1987-09

Horizontal estimation and information fusion in multitarget and multisensor environments.

Fahmy, Alaa Eldin M.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

HORIZONTAL ESTIMATION AND
INFORMATION FUSION IN MULTITARGET AND
MULTISENSOR ENVIRONMENTS

by

Alaa Eldin M. Fahmy
September 1987

Thesis Advisor: Harold A. Titus

Approved for public release; distribution is unlimited

URITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION ICLASSIFIED	1b RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
DECLASSIFICATION / DOWNGRADING SCHEDULE	
PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION val Postgraduate School	6b OFFICE SYMBOL (If applicable) 62
7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
ADDRESS (City, State, and ZIP Code) nterey, CA 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
NAME OF FUNDING / SPONSORING ORGANIZATION	8b OFFICE SYMBOL (If applicable)
9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

TITLE (Include Security Classification)

RIZONTAL ESTIMATION AND INFORMATION FUSION IN MULTITARGET AND MULTISENSOR ENVIRONMENTS

PERSONAL AUTHOR(S)

aa Eldin M. Fahmy

TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year Month Day)	15 PAGE COUNT
.D. Dissertation	FROM ____ TO ____	1987 September	227

SUPPLEMENTARY NOTATION

COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Horizontal Estimation; Multitarget Track Fusion; Rule-based Track Fusion; Knowledge-based Track Fusion; Distributed Estimation in DSN

ABSTRACT (Continue on reverse if necessary and identify by block number)

In recent years, there has been a considerable increase in both the variety and num-r of sensors which needed to be tied together. A new distributed estimation archite-ure for Distributed Sensors Networks (DSN) is introduced. It is called Horizontal timation Architecture (HEA). The term horizontal is used to imply that the geo-aphically dispersed nodes do not differ in rank and are peer-to-peer coupled. Each de is connected by a data link to its neighbors (where possible), thus providing a sh network topology. The introduced HEA has four major components, the local esti-tor, the information fusion process (both together are called a horizontal estimator), e network access protocol, and the controller-decisionmaker.

The HEA techniques are applied to the solution of Multitarget and Multisensor Track-g (MMT) problems in Track-While-Scan (TWS) systems with an emphasis towards track sion. A mathematical framework which encompasses the components of the horizontal

STRIBUTION / AVAILABILITY OF ABSTRACT NCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
NAME OF RESPONSIBLE INDIVIDUAL fessor Harold A. Titus	22b TELEPHONE (Include Area Code) (408) 646-2560 / 22c OFFICE SYMBOL 62Ts

DRM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

estimator is developed, with an emphasis towards the track fusion algorithm. An artificial intelligence approach using expert systems for track fusion has been presented. Through this HEA application its main features and practical usefulness are addressed.

Approved for public release; distribution is unlimited.

Horizontal Estimation and Information Fusion in Multitarget and Multisensor Environments

by

Alaa Eldin M. Fahmy
Colonel, Egyptian Air Force
B.S.E.E., Military Technical College, Cairo, Egypt, 1968
M.S.E.E., Mansoura University, Egypt, 1981

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1987

ABSTRACT

In recent years, there has been a considerable increase in both the variety and number of sensors which needed to be tied together. A new distributed estimation architecture for Distributed Sensors Networks (DSN) is introduced. It is called Horizontal Estimation Architecture (HEA). The term horizontal is used to imply that the geographically dispersed nodes do not differ in rank and are peer-to-peer coupled. Each node is connected by a data link to its neighbors (where possible), thus providing a mesh network topology. The introduced HEA has four major components, the local estimator, the information fusion process (both together are called a horizontal estimator), the network access protocol, and the controller-decisionmaker.

The HEA techniques are applied to the solution of Multitarget and Multisensor Tracking (MMT) problems in Track-While-Scan (TWS) systems with an emphasis towards track fusion. A mathematical framework which encompasses the components of the horizontal estimator is developed, with an emphasis towards the track fusion algorithm. An artificial intelligence approach using expert systems for track fusion has been presented. Through this HEA application its main features and practical usefulness are addressed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENT

Among the many people who have contributed to the success of this research, I would like to first express my deepest gratitude to Professor Harold A. Titus, my dissertation supervisor, for his constant encouragement, professional guidance, and enthusiasm throughout the course of this study. Professor Titus's incessant urging for better understanding of the results obtained has often provided needed inspirations.

Special thanks are due to Distinguished Professor G. J. Thaler, Professor R. Panholzer, Professor N. F. Schneidewind, and Professor M. F. Platzer, for serving as members of the Ph.D committee and for providing valuable comments. The constant encouragement of Distinguished Professor G. J. Thaler is highly appreciated.

The financial and moral support from the Egyptian Government, and the privilege to pursue graduate study at Naval Postgraduate School are greatly appreciated.

Finally, and perhaps most importantly, I thank my wife, Azza, for her love, patience, understanding, and consistent encouragement and for keeping me sane, especially during the last few months. Special thanks to my children, Hany, Ahmad, Mohammad, and Hebba for their understanding and endurance during this work.

I. INTRODUCTION

In recent years, there has been an increasing interest in Distributed Sensors Networks (DSN). Examples can be found in Air Traffic Control (ATC) systems, surveillance systems, and air defence systems. In these systems, computers are sited essentially at the sensor sites, or in the display system, and in the command and control areas. As a consequence, a distributed sensors network implies a computer network, which ensures performance of data processing, organization of information display, and the communication between the different network components, in addition to an estimation and decision making processes.

A. MAIN DSN DEVELOPMENT ISSUES

There are many different issues which arise in the development of a DSN. The fundamental issues are :

1. The design of multisensor architectures.
2. How much local processing capability should a sensor have, and how should the data communicated by a sensor be summarized and compressed?
3. The procedures for data fusion.
4. The performance evaluation of a multisensor configuration.
5. The on-line management/control of an implemented multisensors network.
6. Provision of software and hardware, bearing in mind the growing, restructuring and reconfiguration capabilities.

These issues are difficult. They become more so in the multitarget environment faced in aircraft, missile, battlefield, and ocean surveillance. In these environments, there is the added problem of data association. That is, given a piece of data, one must determine to what it should be attributed. Should it be one of the current objects being considered, and if so, which? or, should it be a new object, or a false alarm?

General architectures of multisensor networks are centralized and distributed. Several problems arise in the analysis and design of the mentioned architectures, among them are:

a. Air-Space management, i.e, allotment of airspace sectors to the sensors of the network.

b. Gathering, routing, management and dissemination of data and results through the communication network.

c. Organization of sensors and processors.

B. REQUIREMENTS FOR DEVELOPMENT OF DSN

There are technological advances in several disciplines, which are providing tools for designers of DSN. An adequate development of DSN requires the successful integration of these disciplines, e.g, modern theory of systems and control, computer science, communications, man-machine interactions, expert systems, information systems, reliability and maintainability. In defense systems especially, there is an increasing interest in simultaneously using various types of sensing techniques co-operating with each other, such as various types of radars, infrared detecting passive radio surveillance, and laser systems. The combination gives rise to a system concept which requires location of objects and detection over a wide frequency spectrum. This decreases the effect of unintentional and

intentional interferences, and increases the system reliability and survivability. In addition, the use of passive sensors improves the system's covertness. Also, a defense system is required to remain reliable and secure in the presence of intense hostile activity, which will probably include destructive actions.

Today, the communication is both slower and more costly than computation and processing. Present trends indicate that these imbalances in speed and costs of communications and processing will not only continue but are likely to greatly increase [Ref. 1]. Military communication systems are generally designed and tested in a peacetime setting. However, their greatest test will come in a conflict situation, when the need to communicate becomes great and hostile enemy actions, including physical destruction and electronic countermeasures will create node and link failures and a dynamically changing network topology. So, one of the prime motivations in DSN is to try to minimize the communication load between different sites or nodes.

Since the sensors are the means by which a decision making process observes the environment , which is generally a dynamically changing environment in the presence of uncertainties, sensor measurements are used to reduce these uncertainties, and determine the current state of the system. So, DSN implicitly includes an estimation process in addition to a decision making process. The future trends of decision making in complex technological environments are towards decentralized strategies. These trends are being supported by the fruitful progress in the direction of

distributed computer architectures, distributed data processing, distributed knowledge based systems and fifth generation computers.

The Japanese Fifth Generation Computer Systems (FGCS) project is directed toward nonnumeric applications. The more traditional systems application of number crunching is not an issue in the FGCS. It is expected that the supercomputers being developed today will perform these tasks. The FGCS project is directed instead toward applied artificial intelligence and symbol manipulation. In other words, the next generation of computers is being designed to manipulate knowledge. It will employ and exploit knowledge-based system technology.

C. DISTRIBUTED ESTIMATION AECHITECTURES

A distributed estimation architecture for DSN is introduced bearing in mind previously mentioned considerations. Research in distributed estimation has progressed along several directions. A survey of these directions can be found in the paper by Chong, Tse and Mori [Ref. 2], and its references [Ref. 3,4,5,6]. The estimation architectures used are mainly hierarchical and centralized. In our approach, the Horizontal Estimation Architecture (HEA) is introduced. The term horizontal is used to imply that the geographically dispersed nodes do not differ in rank. They are of equal status and peer- to - peer coupled, and emphasizing the nonhierarchical architecture used.

The practical usefulness and the performance of HEA techiques are better described with reference to the particular applications for which they are designed. Multitarget Multisensor Tracking (MMT) problems in Track-While-Scan (TWS) systems, has been selected as a specific problem to be

addressed, emphasising the information (track) fusion process. A mathematical model for MMT is presented. An algorithm for pairwise track fusion is derived and the optimality of HEA track fusion is justified. An artificial intelligence approach using expert systems for track fusion has been presented. Through the HEA application its main features and practical usefulness are addressed.

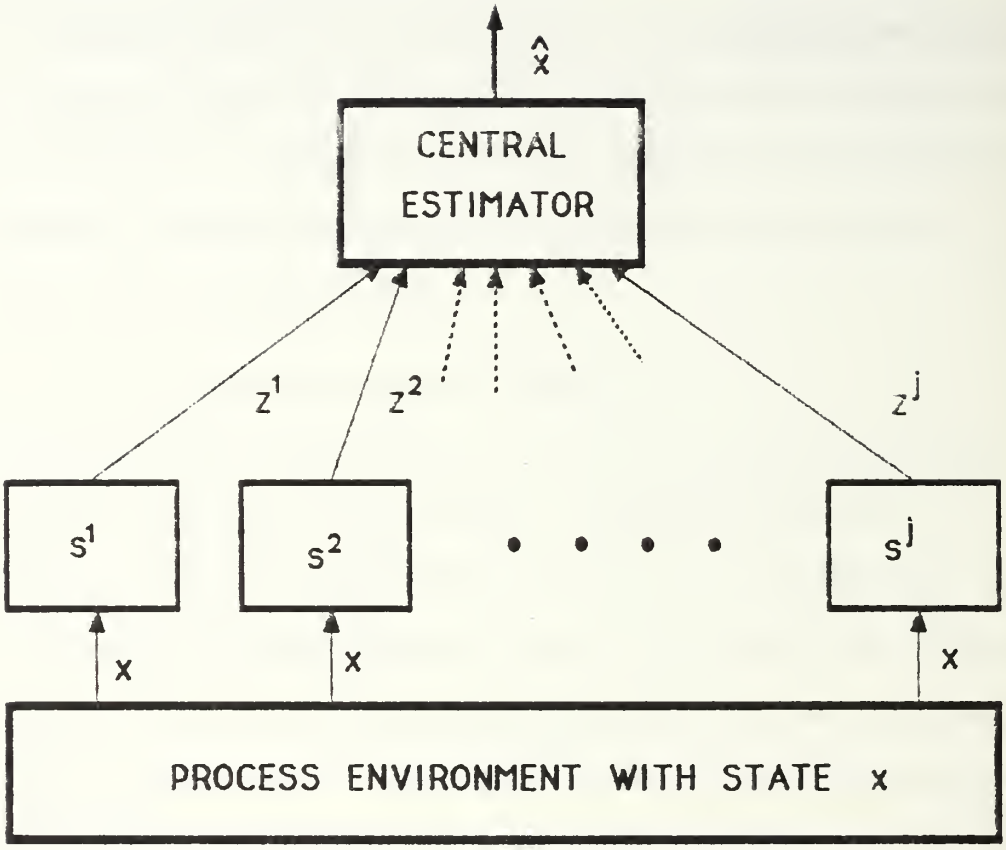


FIGURE 1.1. Centralized Estimation Architecture

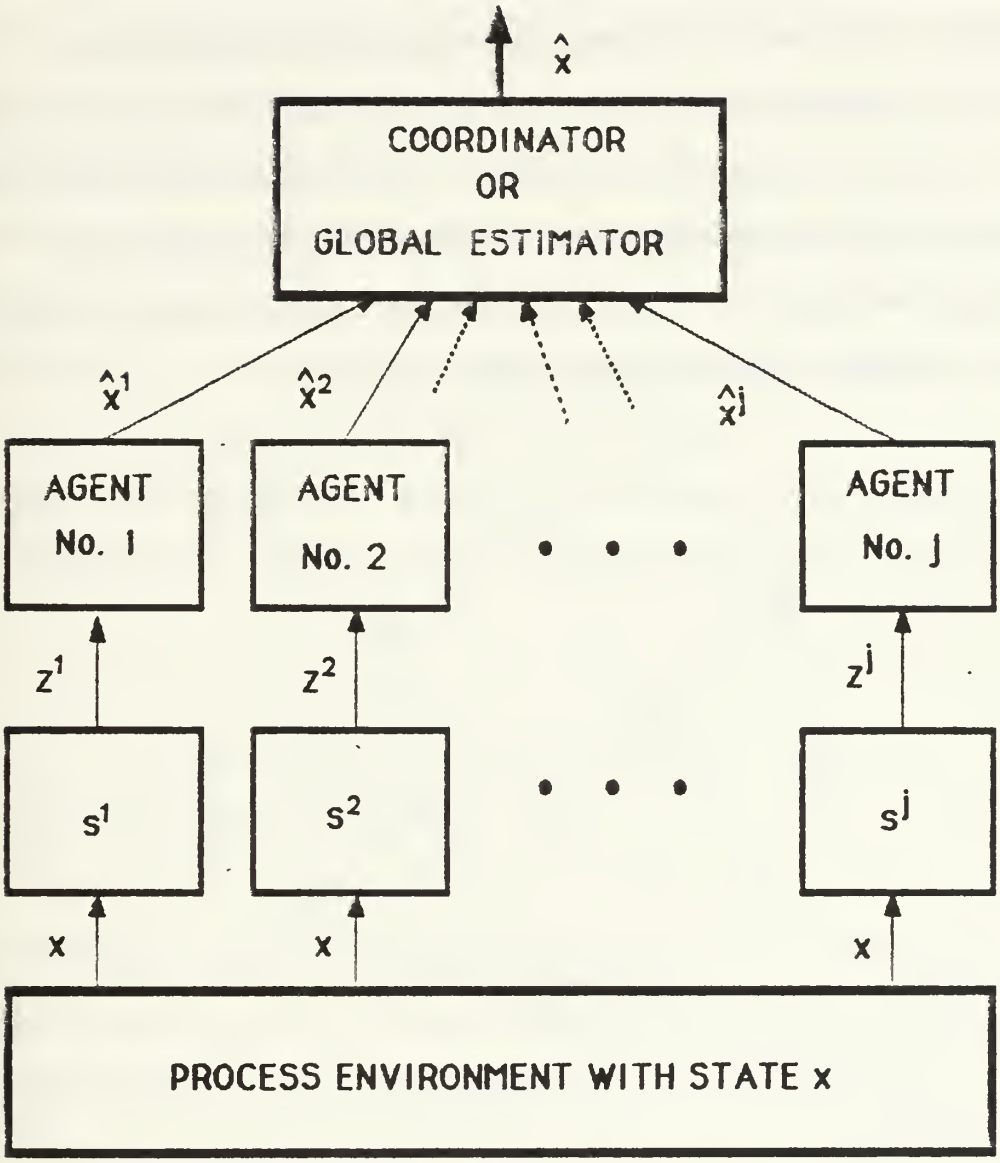


FIGURE 1.2. Hierarchical Estimation Architecture

II. HORIZONTAL ESTIMATION ARCHITECTURE

A. MAIN FEATURES OF HEA

The main concept of distributed estimation applied to DSN is used. Specifically, each node in the network performs local processing of the data collected only by its own sensor or sensors. The locally processed data from each node are communicated through an appropriate data link to its neighboring nodes (where possible), in addition to using it locally. Thus providing a mesh network topology (Figure 2.1).

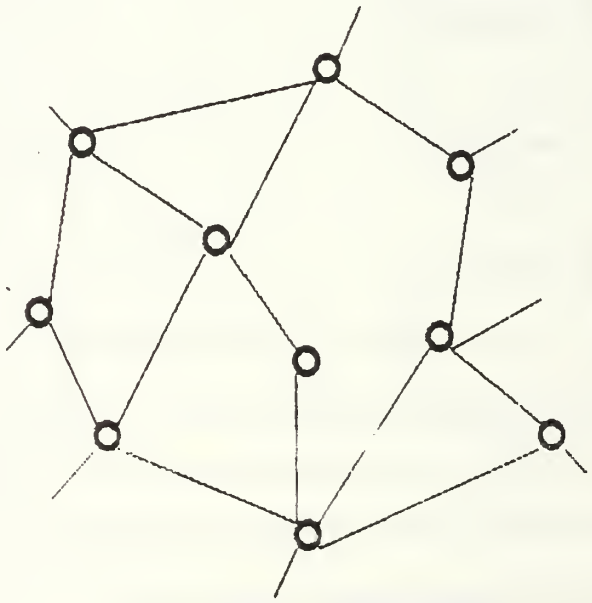


FIGURE 2.1. Mesh Network Topology

B. GRAPH REPRESENTATION OF A MESH NETWORK

Using graph theory with some modifications, we represent the mesh network as a directed graph G [Ref. 7:pp. 424-473].

Definition: Let V be a finite set. A mesh network can be represented as a directed graph (or digraph) G on V, made up of the elements of V, called the nodes of G, and a subset E of the cross product $V \times V$, called the communication links of G. If $a,b \in V$ and $(a,b) \in E$, then there is a communication link from a to b. Node a is called source of the link, with b the terminus, or terminating node, and we say that b is neighbor to a, and a is neighbor to b. It is asumed that G is loop free, and there are no isolated nodes. Generally each node is a source and terminus.

Based on that definition, let V be the set of nodes, while E is the set of communication links, and it is a subset of the cross product of the sets V and V, i.e

$$E \subset (V \times V) \tag{2.1}$$

where $$V \times V = \{ (a,b,c, \ldots\ldots,n) \mid a,b,c,\ldots\ldots,n \in V \} \tag{2.2}$$

and we write the mesh network G as

$$G = (V, E) \tag{2.3}$$

where G, usually is not fully connected, i.e. for all $x,y \in V$, $x \neq y$, there is not necessary a link from x to y. In mesh networks, a link is often directed in both directions (bidirectional). Consequently, if G is a directed graph and $a,b \in V$, $a \neq b$, with both $(a,b), (b,a) \in E$, the undirected link in Figure 2.2(b), are used to represent the bidirectional link shown in Figure 2.2(a). In this case, a and b are called neighboring nodes, and it is represented as

{[a,b]} = {(a,b),(b,a)}. In HEA the data processing capacity can vary from node to node, and communication capacity can vary from link to link.

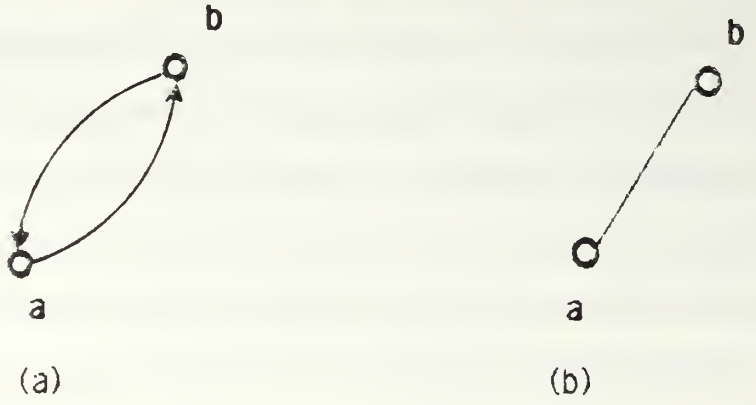
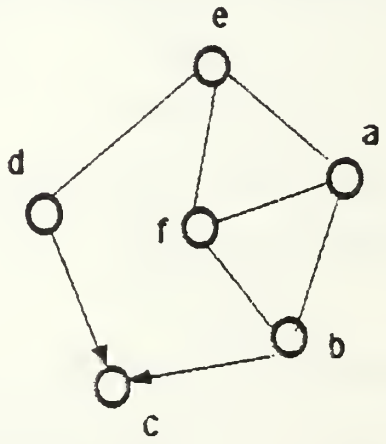


Figure 2.2. Representation of a Bidirectional Link



V = { a,b,c,d,e,f }

E = { [a,b], [a,e], [a,f], [b,f], [e,f], (b,c), (d,c), [d,e] }

G = (V,E)

Figure 2.3. Mesh Network Representation Using Directed Graphs

Example: The mesh network shown in Figure 2.3, is represented as

$$G = (V, E)$$

where $V = \{ a,b,c,d,e,f \}$

and $V \times V = \{ (a,b,c,d,e,f) \mid a,b,c,d,e,f \in V \}$

and $E \subset (V \times V)$

$E = \{ [a,b], [a,e], [a,f], [b,f], [e,f], (b,c), (d,c), [d,e] \}$

It is clear that (b,c) indicate a unidirectional link from b to c, and [a,b] indicate a bidirectional link between a and b.

C. PROPOSED DSN INTELLECTUAL ASSETS

In the literature, data and information are used interchangeably. Actually, data and information are not synonymous. There is a distinction between them. Data become information only when they are useful and available. This means that information is produced as output of data processing operations and used to enhance understanding and to achieve specific purposes [Ref. 8]. Figure 2.4, shows a pictorial view of a proposed DSN intellectual assets shown as a four layer pyramid. The bottom layer is the data collected by sensors, which are processed using a data processing mechanism. The results of the data processing are information, which is the second layer, and it is apparently the processed data. The information added to the expertise of a human expert, resulting in knowledge, which is the third layer. The knowledge is used for developing the knowledge-base, which is the core of the expert system. The expert system is used for aiding the controller-decisionmaker in giving intelligent decisions, which is the utmost purpose of the DSN. It is noted that the human expert is kept in the

loop from time to time to adapt the knowledge-base to cope with the changes in any situation affecting the environment sensed by the sensors.

Based on that, the result of processing data at each node taken by its own sensor or sensors is local information, which is tied or fused with the incoming information from neighboring nodes to obtain an updated situation assesment at each node and forming a coherent picture that resembles as closely as possible what is happening in the local area of interest. Consequently, we have termed the bringing together of locally processed data from different neighboring nodes into a coherent picture " information fusion".

In fact, by information fusion we also mean to include all sources of information, not just that from processing electromagnetic, acoustic, optical and infrared sensors data. There are for example in defense systems, human observers providing intelligence information and a background of encyclopaedic information and operational plans [Ref. 9]. Also, fusing local information and incoming information from neighboring nodes can be done only after deciding that they are originated only from the same origin. So, the fusion process implies a decision process.

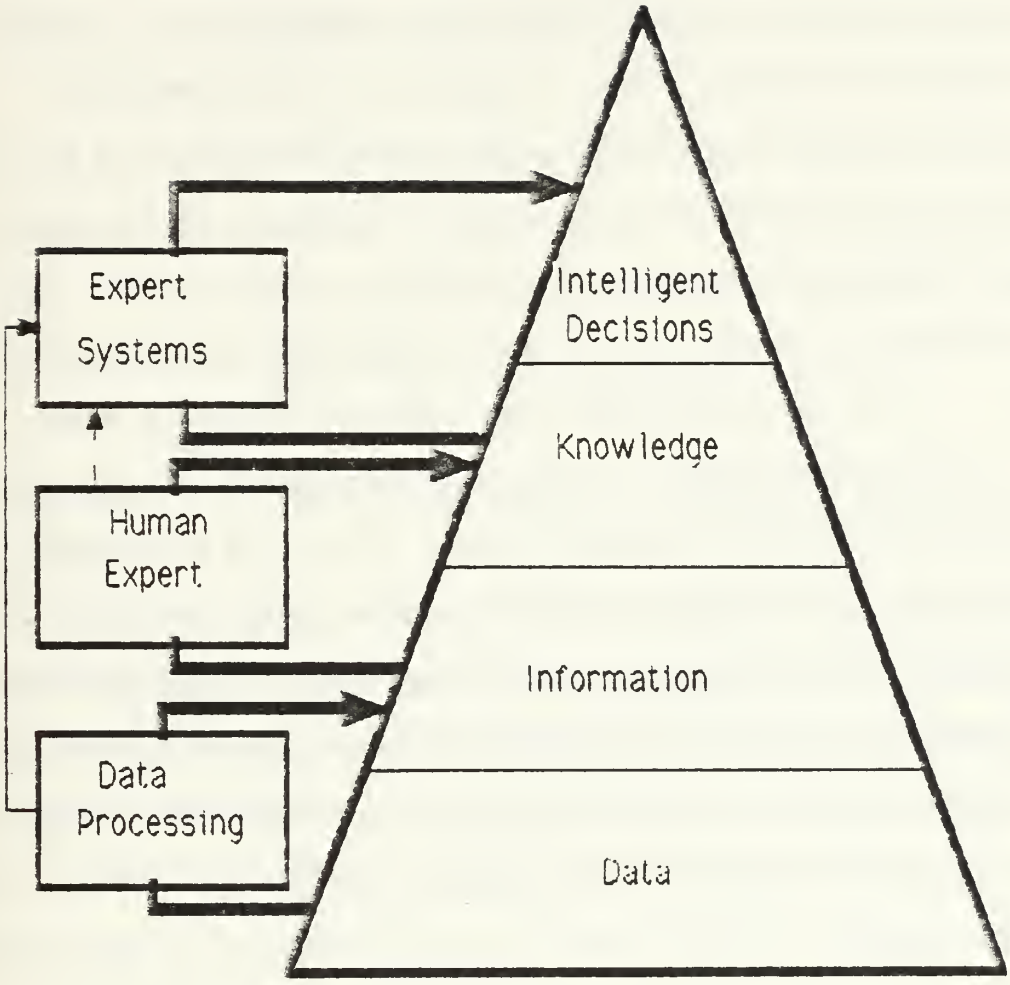


Figure 2.4. Pictorial View of a Proposed
DSN Intellectual Assets

D. INFORMATION ELEMENTS OVERLAPPING IN HEA

The principle of inclusion and exclusion is adopted with slight modifications to represent the information overlapping of different nodes in HEA [Ref. 7:pp. 190-213]. Let S be a set with N=|S|, where |S| is the cardinality or size of S, i.e. denotes the number of elements of S (in MMT case it can be the number of tracks). Let $c_1, c_2, ...,c_t$ be a collection of conditions or properties satisfied by some, or all, of the elements of S. Some elements of S may satisfy more than one of the conditions (e.g. range and bearing in MMT case), while some may not satisfy any of them. For $1 \leq i \leq t$, $N(c_i)$ will denote the number of elements in S that satisfy condition c_i. For $i, j \in \{1,2,3,.....,t\}$, $i \neq j$, $N(c_ic_j)$ will denote the number of elements in S that satisfy both of the conditions c_i, c_j. $N(c_ic_j)$ does not count the elements of S that satisfy only one of the conditions c_i and c_j. If $1 \leq i, j, k \leq t$ are three distinct integers, then $N(c_ic_jc_k)$ denotes the number of elements in S satisfying each of the conditions c_i, c_j and c_k (e.g. range, bearing and velocity in MMT case). For $1 \leq i \leq t$, $N(\bar{c}_i)$ denotes the number of elements in S that do not satisfy condition c_i. In this case, $N(\bar{c}_i) = N - N(c_i)$. If $1 \leq i, j \leq t$, $i \neq j$, $N(\bar{c}_i\bar{c}_j)$ = the number of elements in S that do not satisfy both of the conditions c_i and c_j.

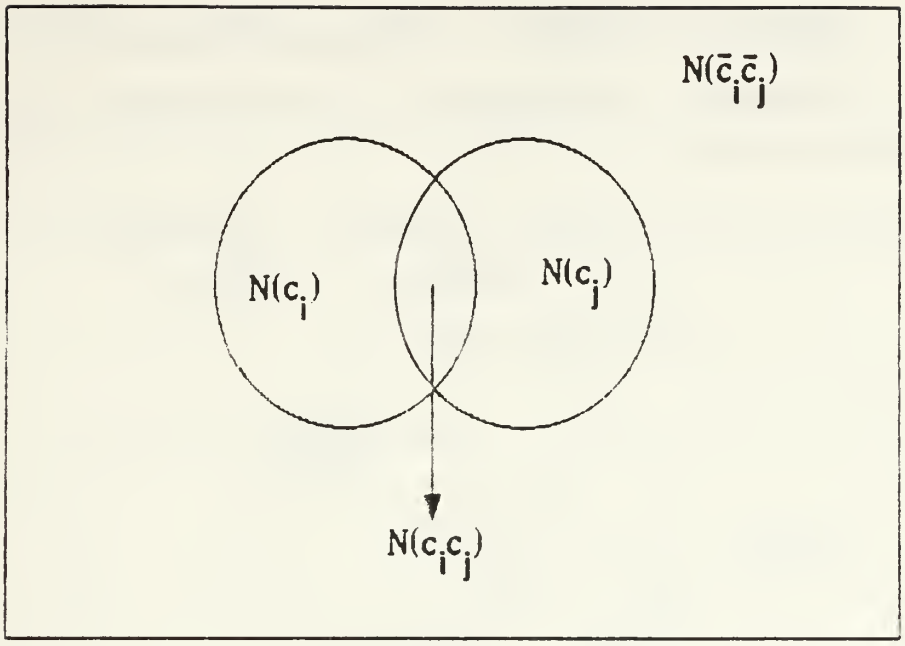


FIGURE 2.5. Two Information Elements Inclusion and Exclusion

From the Venn diagram in Figure 2.5, it is seen that if $N(c_i)$ denotes the number of elements in the left-hand circle and $N(c_j)$ denotes the number of elements in the right-hand circle, then $N(c_ic_j)$ is the number of elements in the overlap, while $N(\bar{c}_i\bar{c}_j)$ counts the elements outside the union of these circles. Consequently, $N(\bar{c}_i\bar{c}_j) = N - [N(c_i) + N(c_j)] + N(c_ic_j)$, where the last term is added on since it was eliminated twice in the term $[N(c_i) + N(c_j)]$. In like manner, from Figure 2.6,

$$N(\bar{c}_i\bar{c}_j\bar{c}_k) = N - [N(c_i) + N(c_j) + N(c_k)] + [N(c_ic_j) + N(c_ic_k) + N(c_jc_k)] - N(c_ic_jc_k)$$

Generally, the number of elements of S that satisfy none of the conditions c_i, $1 \leq i \leq t$, is

$$N(\bar{c}_1\bar{c}_2\bar{c}_3.....\bar{c}_t) = N - [N(c_1) + N(c_2) + N(c_3) + \cdots\cdots\cdots + N(c_t)]$$
$$+ [N(c_1c_2) + N(c_1c_3) + \cdots\cdots + N(c_1c_t) + N(c_2c_3) + \cdots + N(c_{t-1}c_t)]$$

$$- [N(c_1c_2c_3) + N(c_1c_2c_4) + \cdots + N(c_1c_2c_t) + N(c_1c_3c_4) + \cdots$$

$$+ N(c_1c_3c_t) + \cdots + N(c_{t-2}c_{t-1}c_t)] + \cdots + (-1)^t N(c_1c_3c_4 \cdots c_t),$$

or

$$N(c_1c_2c_3.....c_t) = N - \sum_{1\le i\le t} N(c_i) + \sum_{1\le i<j\le t} N(c_ic_j) - \sum_{1\le i<j<k\le t} N(c_ic_jc_k) + \cdots$$

$$+ (-1)^t N(c_1c_3c_4 \cdots c_t) \qquad (2.4)$$

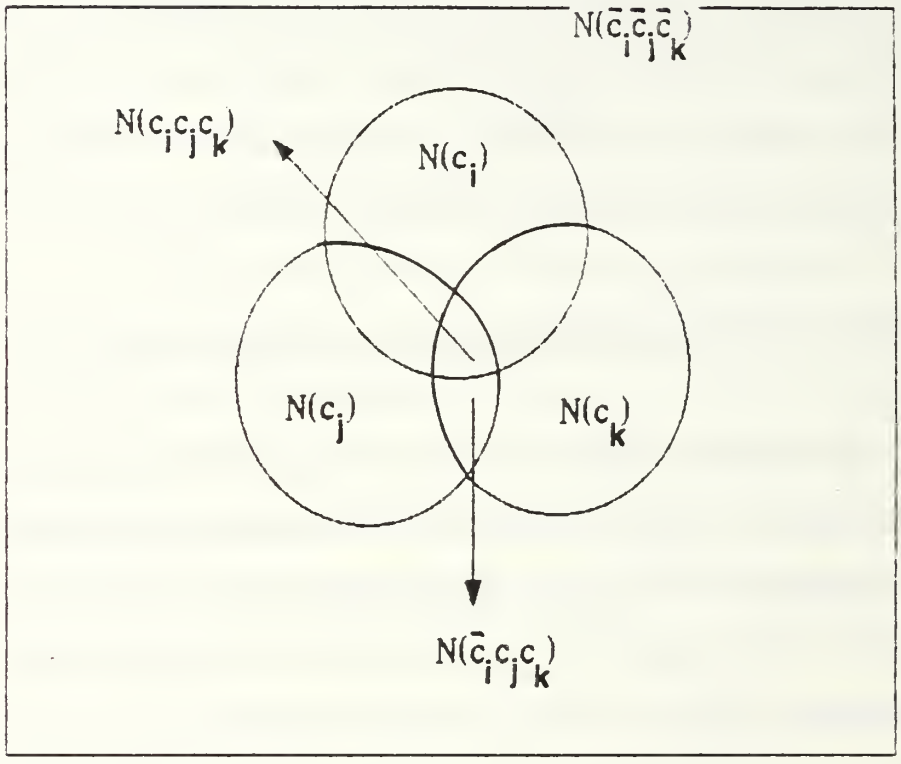


FIGURE 2.6. Three Information Elements Inclusion and Exclusion

Now, assuming the number of elements in S that satisfy exactly m of the t conditions is e_m where $1 \le m \le t$, then

$$e_1 = N(c_1\bar{c}_2\bar{c}_3.....\bar{c}_t) + N(\bar{c}_1c_2\bar{c}_3.....\bar{c}_t) + \cdots + N(\bar{c}_1\bar{c}_2\bar{c}_3.....c_t), \qquad (2.5)$$

and

$$\mathbf{e}_2 = N(c_1c_2\bar{c}_3.....\bar{c}_t) + N(c_1\bar{c}_2c_3.....\bar{c}_t) + ... + N(\bar{c}_1\bar{c}_2\bar{c}_3.....\bar{c}_{t-2}c_{t-1}c_t), \quad (2.6)$$

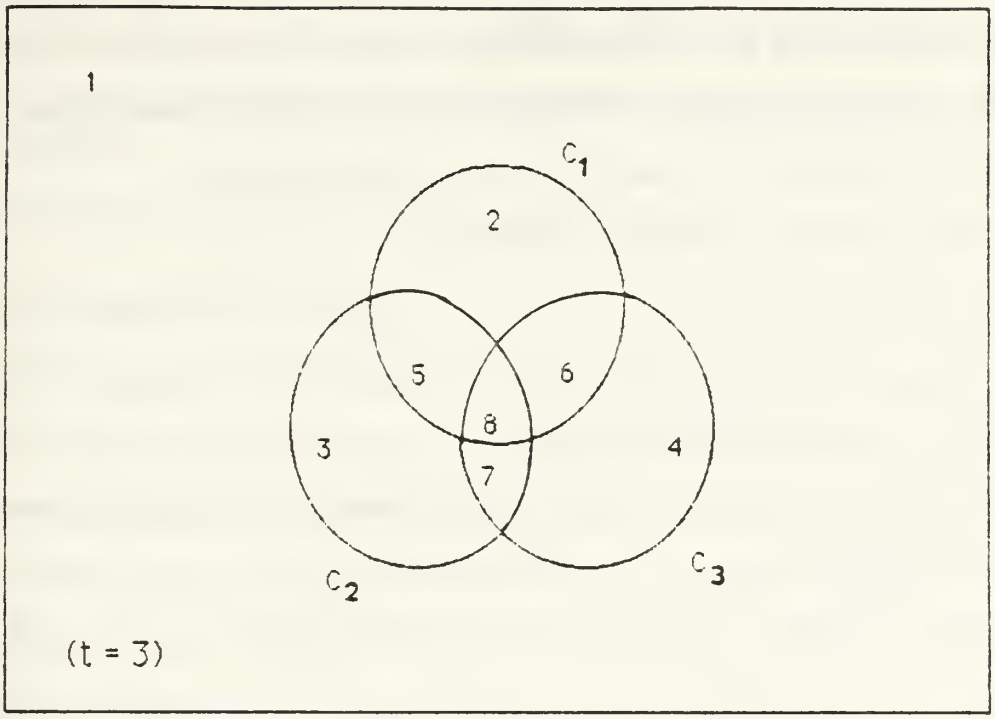


FIGURE 2.7. Information Elements Inclusion for t=3

Using these results as a starting place, as Figure 2.7 is examined, where t=3, a numbered condition is placed beside the circle representing those elements of S satisfying that particular condition. Then $\mathbf{e}_1$ equals the number of elements in region 2, 3, 4. But it can be written that

$$\mathbf{e}_1 = N(c_1) + N(c_2) + N(c_3) - 2[N(c_1c_2) + N(c_1c_3) + N(c_2c_3)] + 3N(c_1c_2c_3) \quad (2.7)$$

In $N(c_1) + N(c_2) + N(c_3)$ the elements in regions 5, 6, and 7 are counted twice and those in region 8 are counted three times. In the next term the elements in regions 5, 6, and 7 are deleted twice. The elements in region 8

are removed six times in $2[N(c_1c_2) + N(c_1c_3) + N(c_2c_3)]$, so the term $3N(c_1c_2c_3)$ is added and end up not counting the elements in region 8 at all. Hence,

$$e_1 = S_1 - 2S_2 + 3S_3 = S_1 - \binom{2}{1}S_2 + \binom{3}{2}S_3 \qquad (2.8)$$

For e_2, the earlier equation indicates that it is needed to count the elements of S in regions 5, 6, and 7. From the Venn diagram

$$e_2 = N(c_1c_2) + N(c_1c_3) + N(c_2c_3) - 3N(c_1c_2c_3)$$

$$= S_2 - 3S_3 = S_2 - \binom{3}{1}S_3 \qquad (2.9)$$

and

$$e_3 = N(c_1c_2c_3) = S_3 \qquad (2.10)$$

Generally, for any $1 \leq m \leq t$, the number of elements in S that satisfy exactly m of the conditions $c_1, c_2, c_3 \cdots\cdots c_t$ is given by

$$e_m = S_m - \binom{m+1}{1}S_{m+1} + \binom{m+2}{2}S_{m+2} - \cdots\cdots + (-1)^{t-m}\binom{t}{t-m}S_t \qquad (2.11)$$

E. COMMUNICATIONS BETWEEN DIFFERENT NODES

To enable communications between different nodes (processors) the following items must be considered [Ref. 10] :

1. The electrical and physical characteristics of the medium chosen for the interconnection.
2. The signalling used to ensure the reliable transmission and reception of data.
3. The means of effecting flow control in order to align the rate of data exchange with the processing capabilities of the machines.

Actually, when two information nodes communicate across a network, a network access protocol between each node and network is needed. The requirements for such a protocol differ significantly for a communication networking technique to another. By protocols, we mean the set of rules, procedures and conventions by which information is transported through the

computer communications network. The key elements of a protocol are [Ref. 11]:

1. Syntax: includes such things as information format and signal levels.
2. Semantics: includes control information for coordination and error handling.
3. Timing: includes speed matching and sequencing.

F. MAIN COMPONENTS OF HEA

Based on the previously mentioned distributed estimation approach, Figure 2.8, shows a pictorial generic view of the main components of the Horizontal Estimation Architecture (HEA). By architecture we mean the set of algorithms, rules, conventions and protocols that implement the horizontal estimation functions and their interrelationships. The introduced HEA has four major components, the local estimator, the information fusion process (both together are called Horizontal Estimator), the network access protocols, and the controller-decisionmaker.

The control policy used is decentralised, with control decisions made independently by each node using the estimates of the states resulting from the information fusion process. The expected value of a quadratic performance index is minimized to assure maximum system robustness. That is, gradual or partial system failures would have minimum degradation in the performance. Also, it avoids the development of complex strategies which are difficult and costly to achieve and implement.

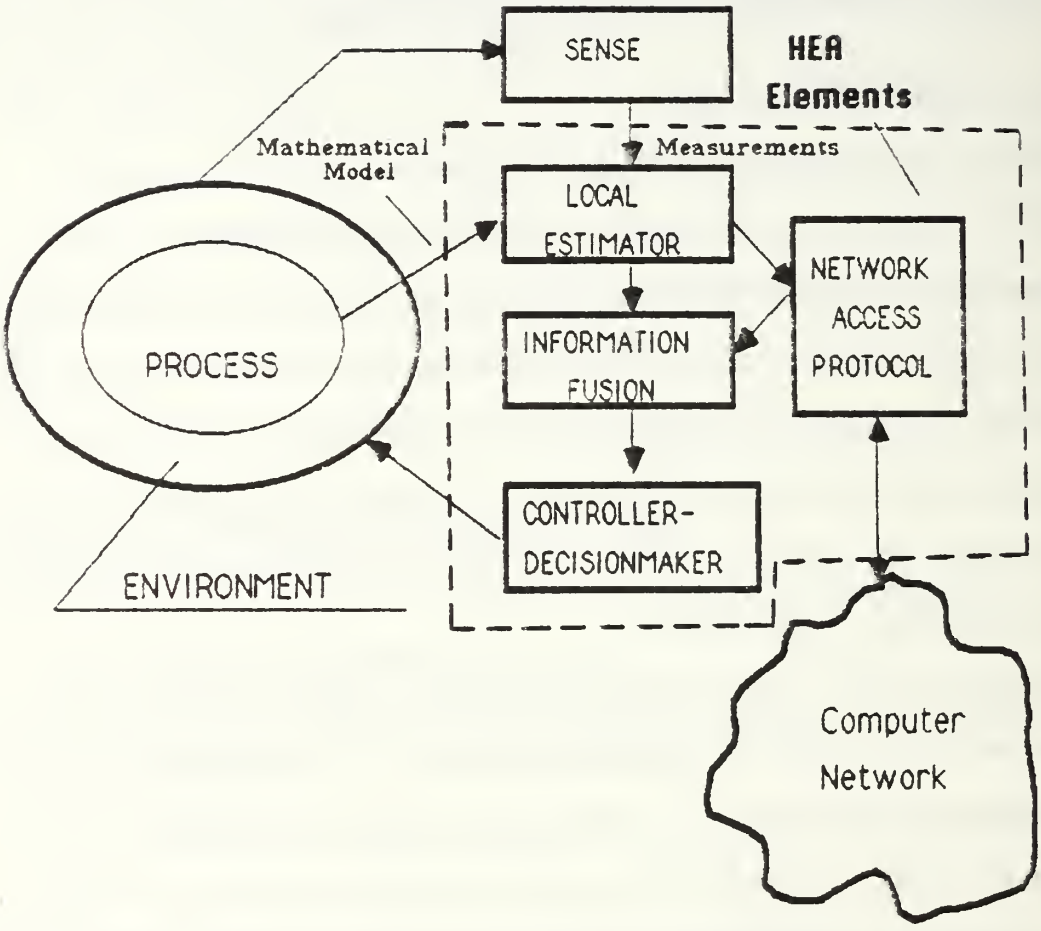
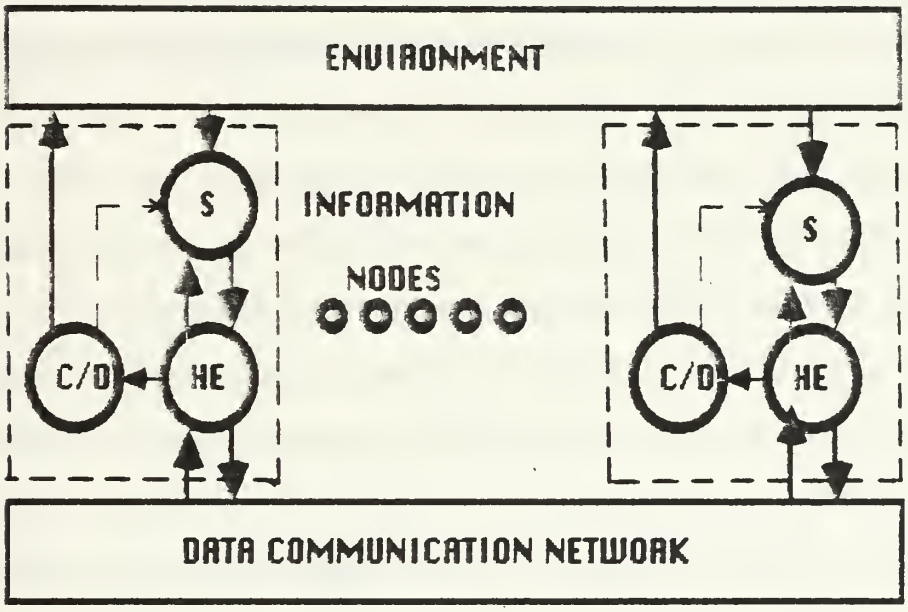


FIGURE 2.8. Pictorial View of HEA
Major Components



S = Sensor

HE = Horizontal Estimator

C/D = Controller/Decisonmaker

FIGURE 2.9. Distributed Sensors Network (DSN)

G. PROPOSED FUNCTIONAL ELEMENTS OF A DSN

Based on the HEA, we view the DSN to be represented as shown in Figure 2.9, by five basic functional elements : the environment, sensors, the horizontal estimator, data communications network, and controller-decisionmaker.

Each sensor complex, horizontal estimator, and controller-decisionmaker is sited in a node, which we call an "information node" to differentiate it from a communication node. Throughout the sequel, node is used to indicate the information node. Each information node is connected through a communications node and an appropriate data link to its nearest neighboring nodes, thus providing a mesh network topology. As mentioned before, each information node performs processing functions by local estimator using the local sensor(sensors) data, communicating the processing results to other neighboring nodes, in addition to using it locally. The information fusion process fuses the information received from neighboring nodes with the local information. The HEA can be further extended to cover a very large scale DSN in the case when each node can be the central processor of a centralised network, and, or a global estimator (parent node) of a hierarchical network (Figure 2.10).

H. MOTIVATIONS TO HEA

In HEA, the approach of "divide and conquer" which is useful for most complex problems can be easily used. Partitioning allows any large complex system to be divided into manageable portions. Another motivation to the HEA is that network splitting and reformation or connection of additional compatible networks are practicable during system operation and do not

cause any restriction as a new system is initiated. Additional advantages of this design approach are, usage of microcomputer systems, which provide a cost-effective solution for data processing, reliability, survivability, local autonomy, heterogeneous feature, low cost communications and its practicability for already existing information nodes, which need to be tied together. The partitioning approach used in HEA allows large complex systems to be divided into manageable proportions.

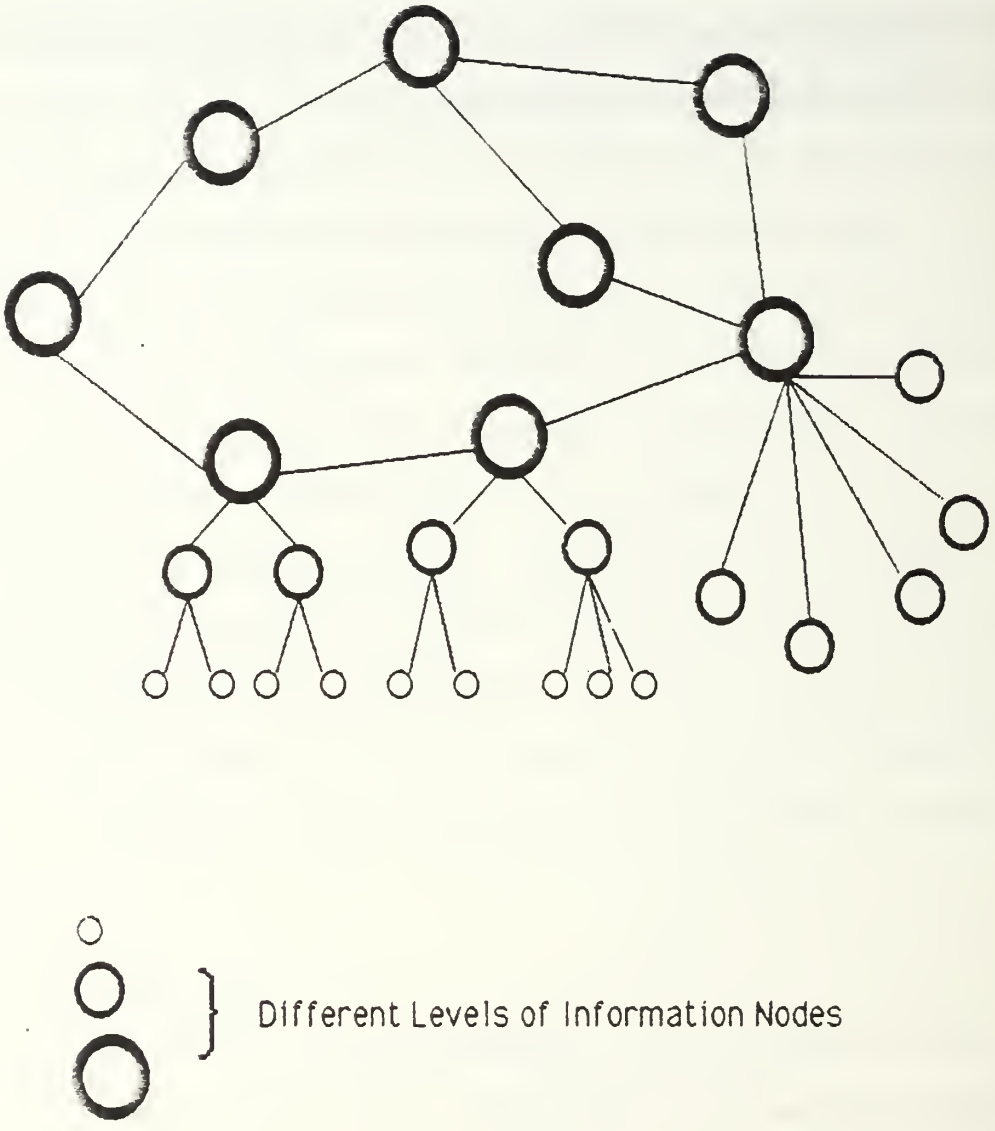


FIGURE 2.10. Very Large Scale DSN

III. OVERVIEW OF TRACK-WHILE-SCAN SYSTEMS

A. TWS RADAR SYSTEM CONCEPT

The process of tracking targets based on discrete radar information obtained while the radar continues to scan the airspace is referred to as the Track-While-Scan (TWS) process[Ref. 12], and is accomplished in a digital computer in modern day surveillance radars. For a TWS system, search and track update functions are simultaneously performed. In TWS, a single sensor scanning at a constant rate illuminates new targets and targets already in track files at the same time. Also, only those target tracks that remain within the TWS search volume can be maintained. In an automatic TWS air-surveillance system, the radar sensor reports measurements of target observations at regular intervals of time to a computer, which then assembles the observations from successive scans into tracks. The computer software must correctly associate new plots with the existing tracks and initiate new tracks from reports received on air targets within range of the radar. The association task is aided by tracking filters which combine noisy measurements with track predictions to obtain smoothed updated track estimates. The predicted position of the target for the next radar scan, based on the smoothed estimate of the current position of the target, is used together with the estimated standard deviation of the prediction to determine the location and size of the region of acceptability of new observations on the target. The tracking filter thus plays an essential role in the function of plot- to- track association, in addition to its role of providing

accurate estimates of the position and motion of the target. In TWS only position and velocity states are estimated because with low update rates (like every 4 seconds), the noisy acceleration estimates would not contribute very much. Typically, in intercept problems, the missile requires only line of sight rates for deployment.

For the TWS system, both search and track update are done simultaneously. At the end of the scan interval, all observations received during the scan are correlated with the existing tracks. An operational air-surveillance system must be capable of tracking many targets simultaneously in an environment that may provide large numbers of false target indications due to real and artificial clutter as well as system noise. The measurement accuracy obtained and the tracking precision of such systems may not demand the most computationally complex filtering operations. It is essential that the filtering operations give sufficient support to the association procedures; any complexity beyond that and not necessary for this task is of diminishing value. It is important however that the filter be sufficiently flexible to adjust quickly to changes in the tracking environment. The literature on the techniques of track filtering is very large and diverse. It is important to use simple, easily implemented, and computationally inexpensive filters which nevertheless retain as far as possible the features of optimality and flexibility which the most general and expensive forms embody.

By automating the target detection and tracking process (Automatic Detection and Tracking) which is called ADT, the bandwidth in the output of a radar is reduced so as to allow the radar data to be transmitted to another

location via narrowband communication links rather than wideband communication links [Ref. 12,13]. This permits the outputs of many radars to be intercommunicated economically.

B. TWS RADAR FUNCTIONS

The signal processor determines the presence or absence of targets while rejecting unwanted signals due to ground clutter, sea clutter, weather, radio-frequency interference, noise sources and intentional jammers. It is performed by coherent and/or non-coherent processing of time samples of the received signals [Ref. 13]. The signal processor is implemented in real-time special purpose hardware. Basic operations that are now routinely exploited as a result of the advances in digital technology [Ref. 14], include:

1. Pulse Compression (PC)
2. Pulse Doppler Processing (PDP)
3. Moving Target Indicator (MTI)
4. Moving Target Detector (MTD)
5. Constant False Alarm Rate (CFAR) circuit

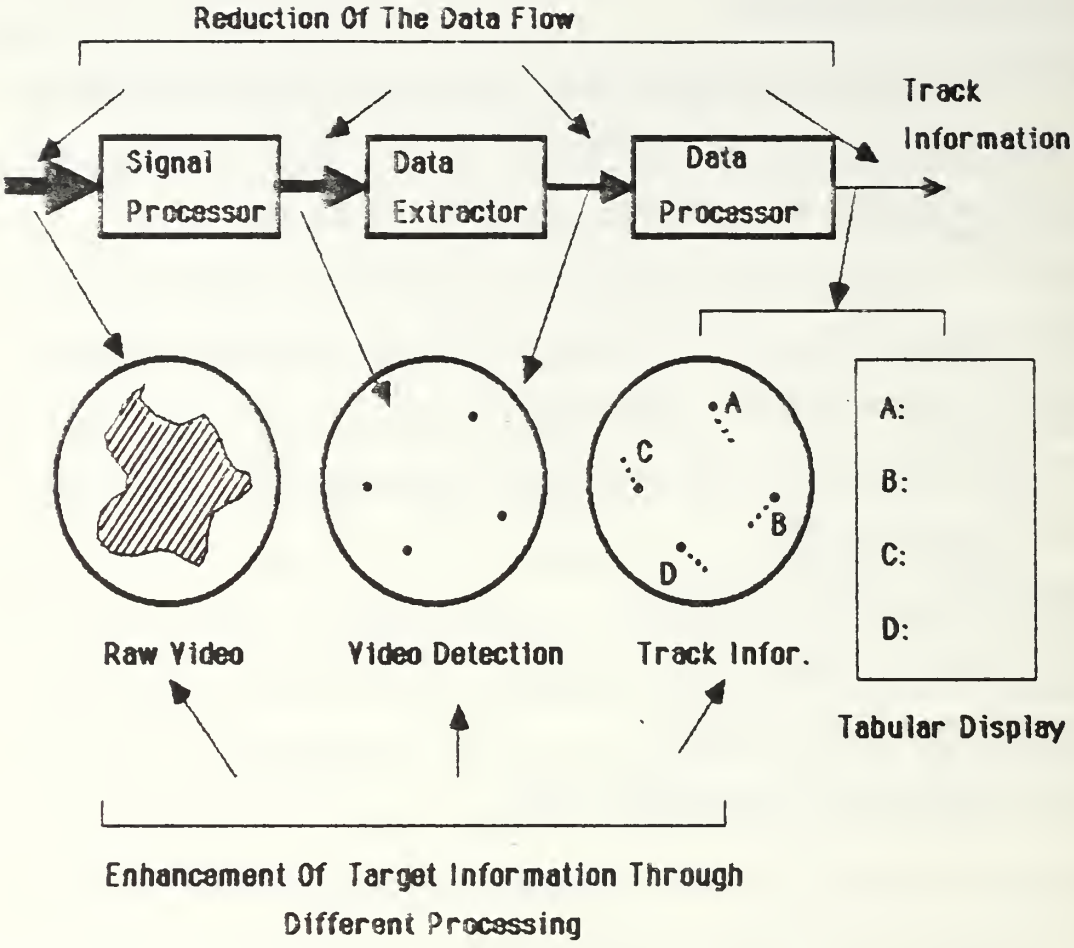


FIGURE 3.1. Outline of Functions Performed in Modern TWS Radar Receiving Phase

The data extractor provides the target measurements in range, angles (azimuth, elevation), radial velocity and possibly provides target signature. In general, a target may cause several detections in adjacent cells in range, Doppler frequency and angles. The centroid of the corresponding pattern of detections gives an estimate of the target measurements. The data extractor is generally implemented with a dedicated microcomputer.

The data processing is performed on a digital computer inserted between the plot extractor and immediately before the display. It can be defined as the set of algorithms which, when applied to the radar detections acquired during successive scans, allows the following:

1. Recognition of a pattern of successive detections as pertaining to the same target.
2. Estimation of the kinematic parameters (position, velocity, and acceleration) of a target, thus establishing a so-called " target track ".
3. Extrapolation of the track parameters.
4. Distinguishing of different targets and thus establishing a different track for each target.
5. Distinguishing of false detections (caused by intentional or natural interference) from true targets.
6. Adaptive refinement of the threshold setting of the signal processor in order to make the radar more or less sensitive in the different spatial directions, depending on the content of a map of false detections refreshed on a scan-to-scan base.

As shown in Figure 3.1, it is important to emphasize that the cascade of signal processor, data extractor, and data processor is ultimately a bandwidth compressor. It receives data at a high rate and processes the signal in such a manner that a relatively low data rate is achieved. This feature is pictorially indicated by the narrowing of the arrows moving from

the left to the right of the cascaded processors. At the same time, there is a progressive discrimination between useful and clutter data, by means of a stepwise decision process. The information handled by the processing chain is progressively manipulated into a form which allows easier decision making by the user. In fact, the raw video signal contains many false echoes. The data extractor isolates the useful target and the data processor identifies the target (possibly labelled with a code), determines the target velocity and additional parameters which are presented in a tabular display. A further observation can be made regarding the increase of the time span in which processing is performed through the cascade. The signal processor involves only a few pulses, the data extractor some adjacent groups of pulses and the data processor consecutive radar scans. In other words, the memory of the processing increases on moving from left to right in Figure 3.1.

So, these TWS systems are excellent candidates for the application of HEA, since they are already have their local estimation processors which perform the functions of track initiation, plot-track correlation, track prediction, track filtering, and track termination.

C. RADAR OUTPUT

The output of a TWS radar is generally a display to visualise the information contained in the radar echo signal in a form suitable for operator interpretation and action. The visualised information on the display is called " synthetic video" in contrast to the so called " raw video" which is the information shown when the display is connected directly to the video output of the receiver [Ref. 14,15]. Synthetic displays, whilst being a processed representation of the signals which are being received by the

radar sensors, have the advantage that all responses, irrespective of the weakness of the returned signal, can be displayed at a constant level of brilliance and clarity. The Plan Position Indicator (PPI), the usual display employed in radar, indicates range and azimuth of a detected target. The idea of tracking is easily visualized if successive scans containing a moving target are superimposed, the target then gives rise to a fairly regularly spaced sequence of returns.

In the past, an operator manually marked the location of the target at each scan with a grease pencil on the face of PPI. This procedure was very simple but offered poor accuracy and simultaneous processing of only a few targets, due to operator fatigue. The limitations of the operator have been overcome by resorting to a computer which automatically performs the whole tracking process. This computer has been referred to as the " data processor " in Figure 3.1. In order to design an automatic procedure to track one or more targets, it is convenient to examine the nature of the plot sequence provided by TWS radar. The better the expected properties of the sequence can be defined, the greater is the ability of the tracker to distinguish among different targets and false plots. False plots are caused by clutter, intentional interference and noise which survives the action of signal processing. The spacing of the consecutive target plots is caused by the target velocity which may vary in time as the target executes various maneuvers. In the case where the target is an aircraft, upper and lower limits can be placed on the magnitude of the velocity. Further, an upper limit on the magnitude of the acceleration of the aircraft usefully restricts the possible tracks the aircraft can make.

D. CLASSES OF TRACKS

In a modern TWS system, a set of computer software establishes and maintains a number of files pertaining to three different classes:

1. Firm tracks; a firm track occurs when the path of a target has been acquired by the data processor and the kinematic parameters estimated with sufficient accuracy.
2. Tentative tracks; a tentative track corresponds to the first phase of the track acquisition of target or clutter plots.
3. Stationary tracks; a stationary track pertains to clutter, since its position does not significantly change from scan to scan.

The files are produced by processing the plots received from the radar on a scan-to-scan basis and stored in a buffer as, as outlined in Figure 3.2. The contents of the output buffers are periodically updated and displayed to the operator. A firm track occurs when the path of a target has been aquired by the data processor and the kinematic parameters estimated with sufficient accuracy. By contrast, a tentative track corresponds to the first phase of the track acquisition of target or clutter plots. Finally, a stationary track pertains to clutter, since its position does not significantly change from scan to scan. The tracks have been divided into different classes because their corresponding processing and utilization differ.

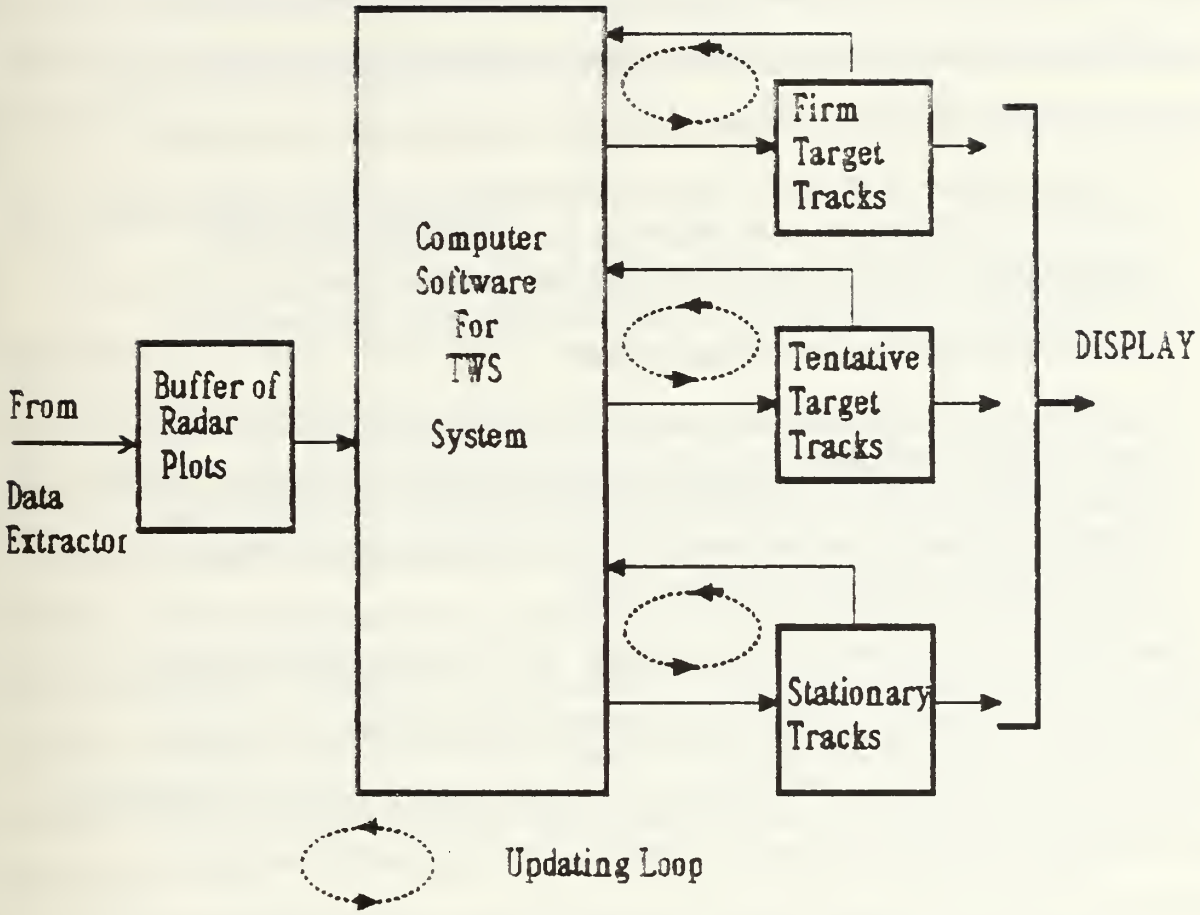


FIGURE 3.2. Track Classes in TWS Systems

E. THE CLUTTER MAP

Because an operational environment may be affected by a large number of false target indications caused by fixed and moving clutter as well as system noise, a particular attention must be paid to filtering out these disturbances. This is carried out by the formation and updating of a clutter map, which identifies spurious plots from the input buffer so that the remaining plots pertain only to valid targets. The use of the clutter map produces some benefits to a radar operating in a clutter environment:

1. It allows proper selection of signal-processor operating modes (e.g. the MTI is switched on only where it is needed, thus avoiding unnecessary detection performance degradation in a clear environment, the thresholds of the detection logic are controlled according to the residual clutter power).
2. It allows removal of clutter points from the radar plot buffer, thus preventing saturation of the processing and the formation of spurious tracks which reduce confidence in the track data. The clutter map updating process is also referred to as a Stationary Track Filter (STF) or Scan to Scan Correlator (SSC), [Ref. 17].

The STF acts as a velocity discriminator upon the plot stream coming from the data extractor. Plots which appear to be stationary or slowly moving from scan to scan are stored in the clutter map. Each new incoming plot is compared with the content of the map, if the plot falls within a defined capture area around one of the clutter map entries, then the new plot is deemed to be clutter. As a consequence , it updates the clutter map content and is deleted from the radar plot buffer. STF can also be considered as a device capable of forming cancellation masks centered around targets (true or spurious) with a velocity below an established threshold value.

The velocity discrimination is performed by comparing the plot displacement in two or more of the subsequent scans with respect to a gate centred around the first position (already measured) of the clutter map entry. It is obvious that, for clutter points or slow-moving targets, the subsequent echoes remain in the gate for a large number of scans, but for fast targets, the echoes are in the gate for only few scans. By computation of the scans required by the target to leave the gate, it is possible to determine its velocity.

F. OPERATIONAL REQUIREMENTS

The operational requirements for data processing vary with the type of application. Typical requirements are estimation accuracy, extrapolation time and system reliability [Ref. 17]. In maritime collision-avoidance systems it is necessary to detect potential collisions well in advance (15-30 minutes) because of long reaction times, especially for large ships such as oil tankers. The estimation error on the forecast position must be lower than one nautical mile. Therefore the accuracy by which the velocity is estimated must be within 1 knot. This requirement determines the degree of filtering to be applied taking into account the measurement error and the data rate of the radar.

In the case of ATC, the radar information may support the function of tactical control, conflict alert and approach control. In tactical control, the radar controller checks the current positions of the aircraft to maintain the standard separations. Whenever the separations tend to be violated, the radar controller advises the pilots involved about the local trajectory modifications required to re-establish acceptable conditions. In conflict

alert, the processing system estimates the forecast positions of all the aircraft to determine the conflicting pairs of aircraft; the extrapolation time is from 1 to 2 minutes. The processing system can also evaluate the trajectory modifications for one or both aircraft to resolve the conflict situation. In this case, one nautical mile may also be assumed as an acceptable accuracy in the estimation of forecast position. Of course, possible maneuvers during the extrapolation time may be taken into account in the evaluation of the conflict area. Therefore, where two aircraft are conflict free when flying straight and could be taken into conflict if one or both maneuver, an order may be given to both aircraft to keep the straight trajectory constant. In approach control, the controller checks that aircraft follow fixed paths to assure a safe landing. A higher accuracy than the preceding one is required here, i.e. a fraction of one nautical mile.

In an air-defence system, the estimated trajectory is generally used to help perform some of the following functions:

a. threat identification
b. threat evaluation
c. calculation of the forecast position (for fire or launch of missile)
d. weapon assignment
e. kill evaluation

The functions a to e do not necessarily use the measurements of one single radar. For example the fire control function c is generally performed by a tracking radar with the characteristics of good accuracy and high data rate.

It may be noted that in ATC, all the trajectories are easy to follow because of path regularity, low acceleration (2-3 m/s^2) and pilot

collaboration; whereas, in air defence, targets have high accelerations (10-50 m/s^2) and manoeuvres that are intentionally evasive. Furthermore, it is important to have a very short reaction time, especially for targets detected at short range such as low-flying aircraft and sea -skimmer missiles. Generally, the data processing from a radar extractor is more difficult for air-defence systems than that for civil-traffic control, because the target acceleration is high and unpredictable. In addition, jammers interfere with the proper working of the system. The effects of these phenomena are reduced by radar signal processing using particular devices, such as moving-target detector and multisidelobe canceller, which limit the false detections.

G. NETTED RADAR SYSTEMS

Much recent interest has centered around radar netting. Estimation of location, velocity and maneuver together with possible identification of each relevant target can be provided by Radar Data Processing (RDP) with an accuracy and reliability greater than that available from a single look radar report. Today, it is also very important to net different types of sensors in addition to radars in order to enhance performance. The current various types of radar networks can be classified according to the level at which the merging of data is taking place into the two following classes:

1. centralized
2. distributed

The centralized architecture (see Figure 3.3) is characterised by the use of a single data processor to which radar plots are transmitted from radar sites. These measurements are processed so as to obtain a single multiradar track for each target. This architecture has the features of :

a. using tracking algorithms with variable data rate.

b. reducing tracking errors because of the higher data rate than with a single radar.

c. requiring more powerful processing resources.

d. being the prevalent type in military applications owing to the higher accuracy gained.

The distributed architecture (see Figure 3.4) is characterised by the use of a computer at each site performing the tracking function on the measurements of a single radar. The monoradar tracks instead of radar plots are then transmitted to a single data-processing centre which combines them in order to establish a single multiradar track for each target. The following features are relevant for this architecture:

- it is capable of local operation.
- it requires computing resources of limited power.
- it mainly requires monoradar processing algorithms.
- it prevails in ATC because of simplicity, reliability and growth capability.

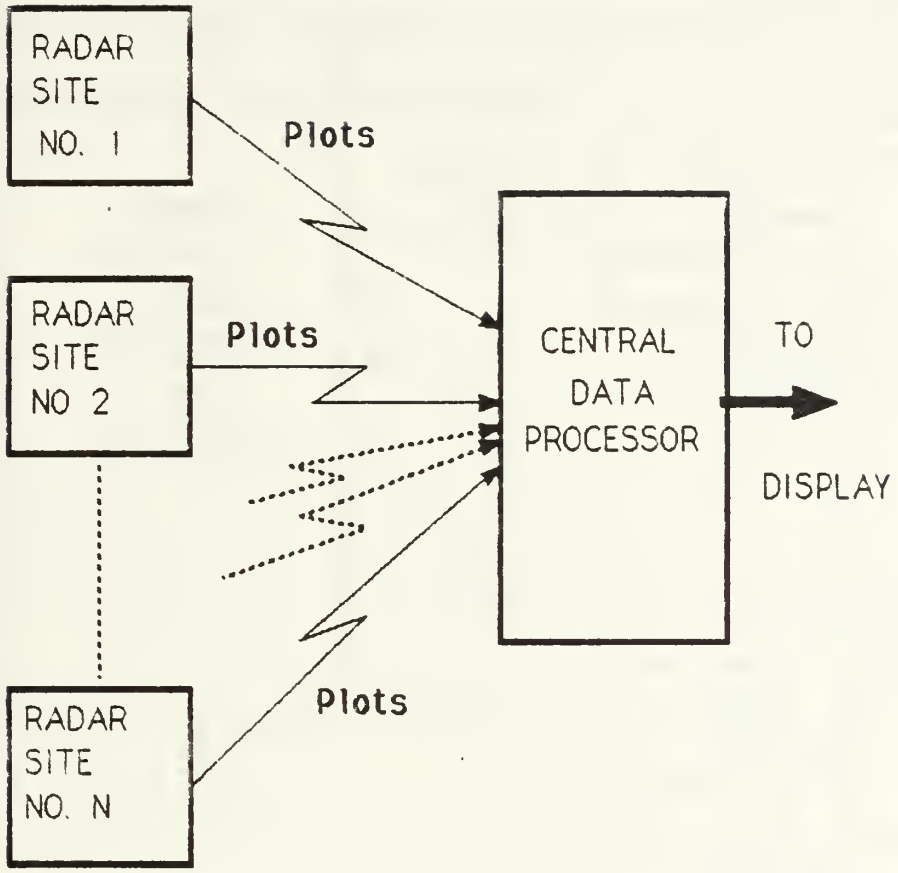


FIGURE 3.3. Centralized Architecture of a Radar Network

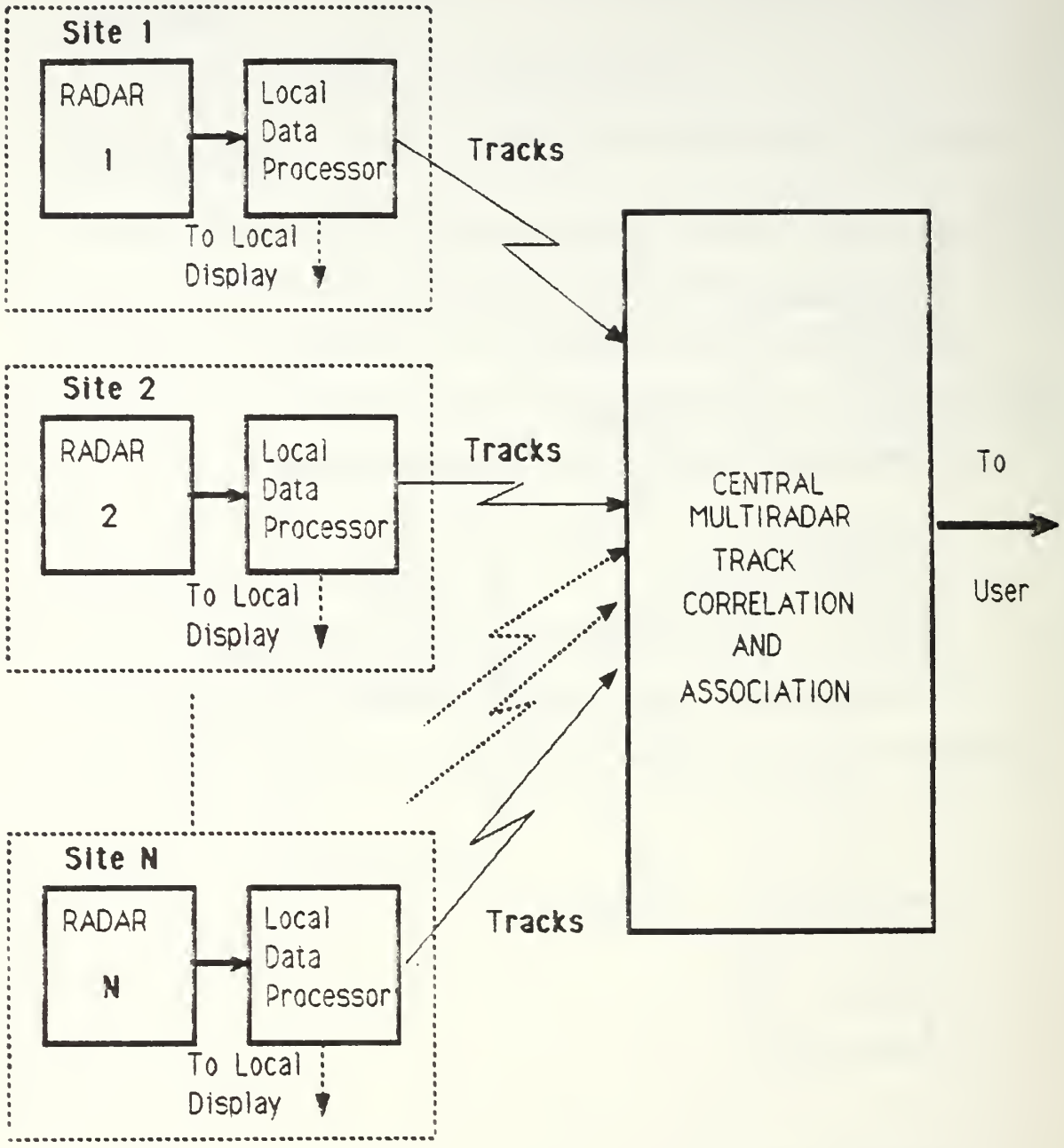


FIGURE 3.4. Distributed Architecture of a Radar Network

The proposed HEA as a distributed architecture applied for radar netting would be convenient and cost effective to be used specially for modern TWS radars which have their local tracking systems already installed and it guarantees the maximum utilization of the local resources of each radar site and taking advantage of the advanced techniques taking place in each major block of the radar systems (Figure 3.5). The problem of the lack of experienced decision makers which lead to the centralization of decision making can be solved by the use of expert systems and encapsulate the decision maker expertise in computer software.

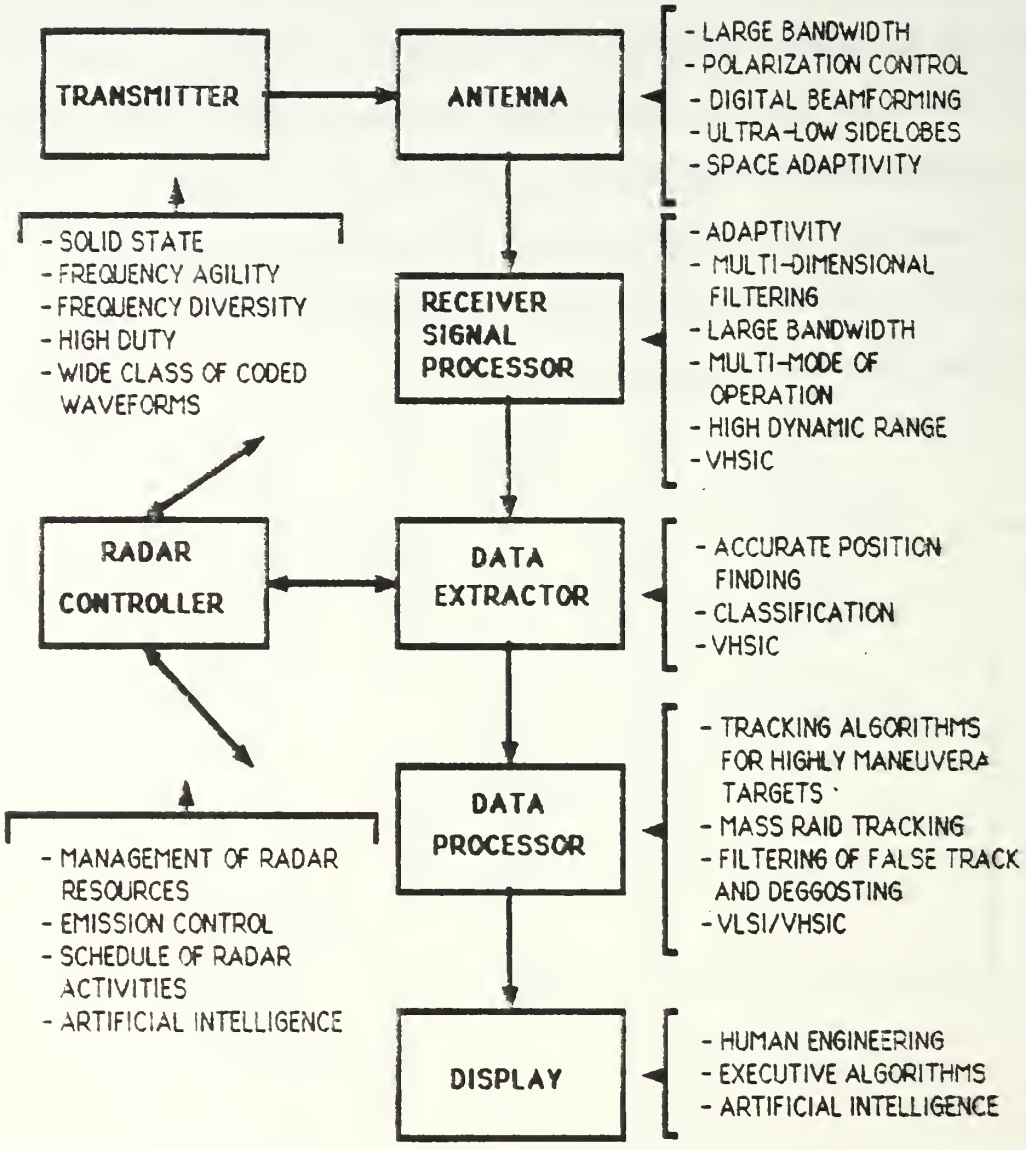


FIGURE 3.5. Advanced Techniques in the Main Blocks of a Radar System

IV. HEA APPLICATION TO MMT PROBLEMS

A. OVERLAPPING COVERAGE OF A RADAR NETWORK

The HEA is applied to the Multitarget Multisensor Tracking (MMT) problems in a tactical air surveillance system. The problem is to produce a statistically meaningful estimate of both the number of the targets present, and of their trajectories. It is assumed that the system employs Track-While-Scan (TWS) radars with different scan rates and with overlapped coverage. The degree of overlap of the radar coverage is defined is as

$$\varepsilon = \sum_{j=1}^{N} A_j \; / \; A_{tot} \qquad (4.1)$$

Where A_j is the area covered by the j^{th} radar

A_{tot} is the total surveillance area controlled by the radar

N is the number of radar sites(nodes)

The parameter ε ranges from 1 (absence of overlap) to N (total overlap). Figure 4.1, shows an example of overlapping coverages between three radars, R_1, R_2, R_3. If the degree of overlap is very small, the advantages of the data redundancy are limited,on average, to small areas and few targets. Assuming that the targets are evenly distributed in A with density δ, then it follows from equation (4.1) that

$$\varepsilon = \sum_{j=1}^{N} n_j \delta \; / \; n_{tot} \delta \qquad (4.2)$$

$$\varepsilon = \sum_{j=1}^{N} n_j \; / \; n_{tot} \qquad (4.3)$$

Where n_j is the number of targets in the area A_j

n_{tot} is the number of targets in the total area A

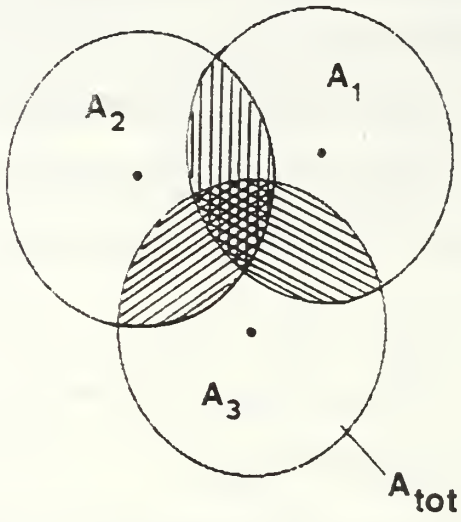


FIGURE 4.1. Example of Overlapping Coverage Between 3 Radars in a Network

Generally, the most obvious exploitation of additional radar sites is to extend the coverage beyond the maximum range of a single radar, as established either by line of sight or by radar power. Also, the viewing of a target from different aspects angles tends to reduce target fades, glint and terrain masking effects. It is assumed that radar sites will be primarily chosen with the aim of optimal radar coverage of the entire network. We consider the multitarget and multisensor problems separately, and then merge the two. This means that the multitarget tracking problem is solved

at the sensor (node) level using the appropriate local estimation algorithm. When the multitarget tracking problem is solved separately for each individual node, a somewhat redundant view of the surveillance area will result, depending on the degree of overlapping coverages between the radars. The output of the local estimator is a group of different tracks. A track, is meant to be a state trajectory estimated from a set of sensor measurements that have been associated with the same target [Ref. 18]. By assuming that the multitarget tracking problem is solved separately for each individual sensor (node), the track estimates for targets in the surveillance area become consolidated first at the level of each individual node. The information fusion process decides whether more than one track from different nodes represent the same target, and combines the corresponding consistent tracks. The techniques for information fusion can be based on algorithmic processing of target kinematic information and heuristic reasoning for attribute information by using expert systems to encapsulate target identity, behaviour, intent, tactical appreciation and human expertise in computer software.

B. DISCRETIZATION OF A SYSTEM DYNAMICS MODEL

The mathematical model of a target motion, as state equation, is derived when assuming that the target normally moves at constant velocity, and turns or evasive maneuvers may be considered as perturbations upon the straight lines. Therefore acceleration is a driving input for the state equation which is usually linear. A simple way to model the unpredictable behaviour of acceleration is to consider a non-Gaussian stochastic stationary process (symmetric with zero mean and proper standard deviation) with a

correlation depending on the time duration of manoeuvre [Ref. 18]. When an aircraft flies, it normally maintains a constant velocity, constant heading trajectory. Consequently, the system dynamics model is given by the following continuous time state equation

$$\dot{x}(t) = A\,x(t) + G\,w(t) \tag{4.4}$$

Where $w(t)$ is the random forcing function (continuous time white process noise), $w(t) \sim (0, Q)$. We wish to put the continuous equation (4.4) into the discrete form

$$x(k+1) = \Phi\,x(k) + w(k) \tag{4.5}$$

Where $w(t)$ is the contribution from the random forcing function. The general solution of (4.4) for $x(t)$ given the value $x(t_0)$ at initial time t_0 is

$$x(t) = \Phi(t,t_0)\,x(t_0) + \int_{t_0}^{t} \Phi(t,\tau)\,G(\tau)\,w(\tau)\,d\tau \tag{4.6}$$

Defining T to be the sampling interval, to describe the state propagation between states, let $t = (k+1)T$, $t_0 = kT$ for an integer k and assuming stationarity, so that:

$$\Phi(t,t_0) = \Phi(t-t_0) = \Phi(T) \tag{4.7}$$

Defining the sampled state function as $x(k) = x(kT)$ we can write

$$x(k+1) = \Phi(T)\,x(k) + \int_{kT}^{(k+1)T} \Phi((k+1)T-\tau)\,G\,w(\tau)\,d\tau \tag{4.8}$$

The second term in the right hand side is a low pass filtered version of the continuous white process noise $w(t)$ weighted by the state transition matrix and the noise input matrix G.

On changing variable twice ($\xi = \tau - kT$ and then Also, $\tau = T - \xi$), the limits of integration can be set to 0 and T, so that, as a first step in the discrete formulation of (4.4), we obtain

$$x(k+1) = \Phi(T)\, x(k) + \int_0^T \Phi(\tau)\, G\, w(\tau)\, d\tau \qquad (4.9)$$

Again under the assumption of stationarity, the transition matrix can be expressed in terms of the matrix exponential. Then, an expansion can be performed to give an approximate solution:

$$\Phi(T) = e^{AT} = I + AT + (AT)^2/2 + \ldots\ldots\ldots\ldots\ldots + (AT)^n/n! \qquad (4.10)$$

Where I is the identity matrix. If terms of order T^2 and higher are disregarded, then the result is Euler's approximation to the sampled system. Equation (4.9) becomes

$$\Phi(T) = I + AT \qquad (4.11)$$

and the discrete - time process noise w(k) relates to the continuous time one as follows

$$w(k) = \int_0^T \Phi(\tau)\, G\, w(\tau)\, d\tau \qquad (4.12)$$

To find the covariance Q (k) of the new noise sequence w(k) in terms of Q, we write

$$Q(k) = E[\, w(k)\, w^T(k)\,]$$

$$Q(k) = \int_{kT}^{(k+1)T}\int \Phi((k+1)T-\tau)\, G\, E[w(\tau)\, w^T(\sigma)]\, G^T \Phi^T((k+1)T-\sigma)\, d\tau\, d\sigma \qquad (4.13)$$

But $E[w(\tau)\, w^T(\sigma)] = Q\, \delta(\tau - \sigma)$

So

$$Q(k) = \int_{kT}^{(k+1)T} \Phi((k+1)T-\tau)\, G\, Q\, G^T\, \Phi^T((k+1)T-\tau)\, d\tau \qquad (4.14)$$

by changing variables twice as before

$$Q(k) = \int_{0}^{T} \Phi(\tau)\, G\, Q\, G^T\, \Phi^T(\tau)\, d\tau \qquad (4.15)$$

It is worth noting that even if Q is diagonal, Q (k) need not be. Sampling can destroy independence among the components of the process noise. Using (4.10) then

$$Q(k) = G\, Q\, G^T\, T + (A\, G\, Q\, G^T + G\, Q\, G^T A^T)T^2 / 2! + \cdots\cdots\cdots \qquad (4.16)$$

In Euler's approximation the process noise covariance Q (k) results from multiplication by T.

$$Q(k) = G\, Q\, G^T T$$

Figure 4.2. shows the sequence of operations used to evolve the discretized equation of an aircraft.

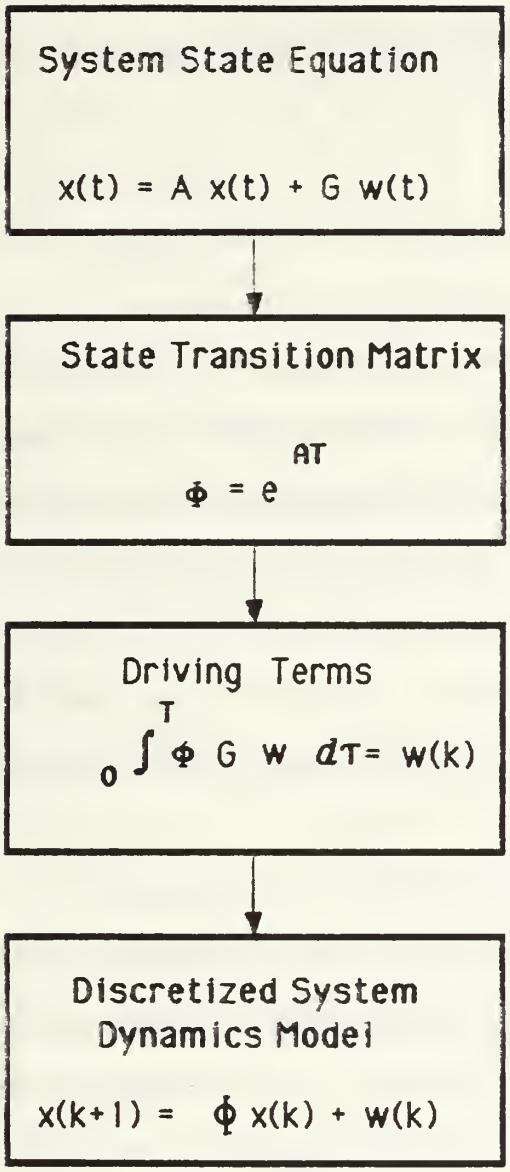


FIGURE 4.2. Evolving of the Discretized System Dynamic Model

C. MMT MATHEMATICAL MODEL

Suppose that s targets are present, and the discretized equations of motion for each target being tracked is adequately modeled by a separate stochastic difference equations of the form

$$x^i(k+1) = \Phi^i(k+1,k)\, x^i(k) + w^i(k) \qquad k = 0,1,2,\ldots\ldots,n \qquad i = 1,2,3,\ldots\ldots,s \qquad (4.17)$$

The superscript indices i denote the various targets. $x^i(k)$ is the $n\times1$ state vector (which generally includes at least position and velocity coordinates) of the tracked target at the k^{th} sample time. $\Phi^i(.,.)$ is the $(n\times n)$ state transition matrix, and $w^i(k)$ is a $p^i\times1$ state excitation vector, which is usually constructed in aircraft tracking applications to account for both maneuvers and modeling errors, and is generally assumed to be white and gaussian, with zero mean and covariance $Q^i(k)$. In a TWS system, the k^{th} sample will occur approximately at time kT^j, where T^j is the scan time of the j^{th} sensor The corresponding discrete measurement vector from each node is given by

$$z^j(k) = h^j(x^i(k),k) + v^j(k) \qquad j = 1,2,\ldots\ldots,N \qquad (4.18)$$

Where the superscript j indicates the various nodes. $z^j(k)$ is a $q^j\times1$ dimensional measurement vector at node j at stage k. $h^j(.,.)$ is the observation equation for j^{th} sensor, $v^j(k)$ is assumed to be white and Gaussian, with zero mean and covariance $R^j(k)$. It is further assumed that $w^i(k)$ and $v^j(k)$ are not serially or cross-dependent, in particular

$$E[\, w^i(k)\, v^j(l)\,] = 0 \qquad \text{for all } k \text{ and } l \qquad (4.19)$$

In the s-target case, each individual measurement $z^j(k)$ of an actual target at time k is drawn from a mixture probability density of the form

s

$$p(z^j(k) \mid x^1(k), x^2(k), \ldots\ldots, x^s(k)) = \sum_{i=1} p_i \; p(z^j(k) \mid x^i(k)) \qquad (4.20)$$

with unknown priors denoted p_i.

Denoting the number of measurements at time k by m(k), a set of $\sum_{k=1}^{n} m(k)$ measurements is

$$Z^j = \{ \{ z_i^j(k) \}_{i=1}^{m(k)} \}_{k=1}^{n} \qquad j = 1,2,\ldots\ldots,N \qquad (4.21)$$

Where Z^j is the cumulative set of measurements at each node.

It is the essence of multitarget tracking problems that there is a large amount of overlap among the component densities of (4.20), so that the target-to-measurement correspondece $x^i(k) \Longleftrightarrow z^j(k)$ is difficult to obtain. The set of all measurements Z^j can be partitioned into m tracks

$$\lambda^i \subset Z^j$$

$$Z^j = \lambda^1 \cup \lambda^2 \cup \lambda^3 \cup \ldots\ldots\ldots \cup \lambda^m$$

where $\lambda^i \cap \lambda^j = 0 \qquad$ for $i \neq j$

One of the tracks (e.g λ^m) contains all of the false alarms. With this notation, any hypothesis concerning measurements made by the surveillance system (the number of targets present, which data points belong to which target) can be defined as a family of subset

$$\lambda^e \subset Z^j$$

The defining characteristic of multitarget tracking problem is that a number of partitions can be found, each one composed of tracks λ^e that appear feasible from the standpoint of the likelihood tests.

D. THE HORIZONTAL ESTIMATOR

Figure 4.3, shows the processing of target data using the Horizontal Estimator (HE) at each node. By using tracking filters such as those discussed in references [Ref. 18, 19, 20, 21], produces local track data bases consisting of the target state vectors and estimation error covariances at specific points of time. Typically, the local track data base will contain kinematic information and sometimes attribute information is included. The typical kinematic information would be :

$\hat{x}^{ij}(k|k)$ -- The target local track estimate at time k using the local sensor data of node j, i.e

$$\hat{x}^{ij} = E[x^i \mid Z^j]$$

$p^{ij}(k|k)$ -- The local error covariance matrix associated with the target local estimate $\hat{x}^{ij}(k|k)$, i.e

$$p^{ij} = E[(x^i - \hat{x}^i)(x^i - \hat{x}^i)^T \mid Z^j]$$

Thus, $(\hat{x}^{ij}, p^{ij})$ records are created by processing the raw sensor for each node with an appropriate filter (local estimator) to produce the local track data base.

Before fusing any two tracks from two different nodes, the two tracks are referred to the same coordinate system using the required transformation, and to the common time instant using the prediction equations of the Kalman filter [Ref. 22].

$$\hat{x}^{ij}(t \mid t_{ij}) = \Phi^i(t \mid t_{ij})\ \hat{x}^{ij}(t_{ij} \mid t_{ij}) \qquad \begin{matrix} i = 1,2 \\ j = 1,2 \end{matrix} \qquad (4.22)$$

$$\text{and} \quad p^{ij}(t \mid t_{ij}) = \Phi^i(t \mid t_{ij})\ p^{ij}(t_{ij} \mid t_{ij}) \qquad \begin{matrix} i = 1,2 \\ j = 1,2 \end{matrix} \qquad (4.23)$$

Where t is the common time instant

t_{ij} is the time of the last updating for the ith track of the jth sensor

ϕ^{i} is the transition matrix of the target model

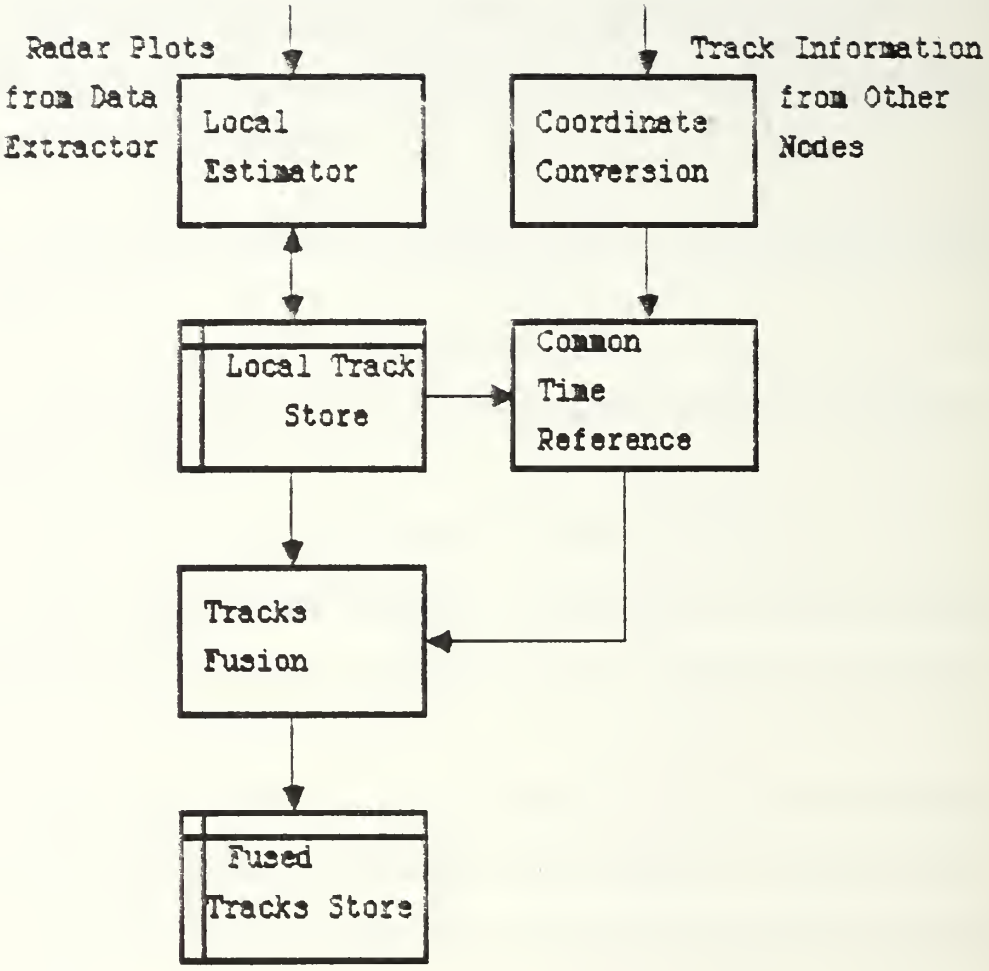


FIGURE 4.3. Processing of Radar Data at Each Node in HEA

V. LOCAL ESTIMATION

A. LOCAL ESTIMATOR FUNCTIONS

Generally, the determination of position and velocity of a target using radar measurements such as range, bearing and range rate, is a problem of nonlinear estimation. In many cases, the relationship between the measured data (e.g, range, azimuth, doppler velocity) and the target dynamic parameters is nonlinear. A rigorous treatment of the nonlinear estimation problem requires the use of stochastic integrals and stochastic differential equations. The optimal (conditional mean) nonlinear estimator cannot be realized with a finite-dimensional implementation, and consequently, all practical nonlinear filters must be suboptimal [Ref. 19].

As shown in Figure 5.1, the local estimator performs the functions of track initiation, plot-track correlation, track prediction, track filtering, and track termination. An overwhelming number of approaches to filtering and prediction for multitarget tracking have been developed recently in response to the ever-increasing importance of the subject. However, at this stage of development, no standard approaches are generally accepted for all applications. A wide variety of techniques have been proposed for many diverse applications, but the multitarget system designer must choose the techniques best suited to his particular problem.

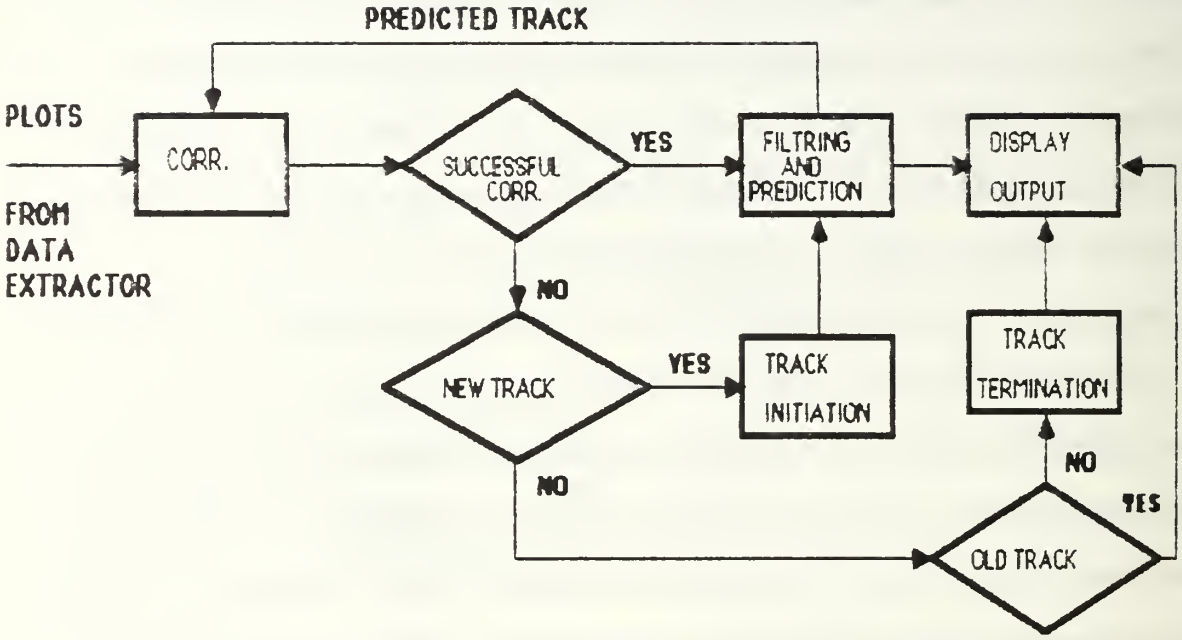


FIGURE 5.1 Basic Functions of Local Estimator

There is a comprehensive set of papers that illustrate commonly known techniques for solving the multitarget tracking problem. Among them is the paper by Reid [Ref. 23], the survey paper by Bar-shalom [Ref. 20], and the paper by Chang [Ref. 19]. There are two most commonly used conventional approaches to filtering and prediction for multitarget tracking [Ref. 18]. The first is to use fixed tracking coefficients, like the $\alpha-\beta$ tracker and the $\alpha-\beta-\gamma$ tracker [Ref. 18, 24, 25, 26] which has computational advantages for systems with large numbers of targets. The second is Kalman filtering which generates time-variable tracking coefficients that are determined by a priori models for the statistics of measurement noise and target dynamics. So, a Kalman Filter (KF) is a computational algorithm that processes measurements to deduce a minimum variance, unbiased error estimate of the state of a system by using the system and measurement dynamics, assumed statistics of system noises and measurement errors, and known initial condition information.

B. CONVENTIONAL KALMAN FILTERING

With expanding computer capabilities, the Kalman filtering is becoming increasingly more appealing to the system designer. The Kalman gain sequence is chosen automatically, based on the assumed target maneuver and measurement noise models, which means that the same filter can be used for varying targets and measurement environments, by changing a few key parameters. Also, the Kalman filter provides a convenient measure of the estimation accuracy through the covariance matrix. This measure is required to perform the gating and correlation functions accurately. In addition, having a measure of the innovation sequence, is useful for

maneuver detection, and upon maneuver detection the Kalman filter model provides a convenient way to adjust for varying target dynamics. The five equations for the conventional Kalman filter are:

$$\hat{x}(k|k-1) = \Phi\,\hat{x}(k-1|k-1) + w \qquad \text{state extrapolation} \qquad (5.1)$$

$$\mathbf{p}(k|k-1) = \Phi\,\mathbf{p}(k-1|k-1)\,\Phi^{T} + Q \qquad \text{error covariance extrapolation} \qquad (5.2)$$

$$K(k) = \mathbf{p}(k|k-1)\,H^{T}\,(H\,\mathbf{p}(k|k-1)\,H^{T} + R)^{-1} \qquad \text{Kalman gain} \qquad (5.3)$$

$$\mathbf{p}(k|k) = (I - KH)\,\mathbf{p}(k|k-1) \qquad \text{error covariance update} \qquad (5.4)$$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K\,(z(k) - H\,\hat{x}(k|k-1)) \qquad \text{state update} \qquad (5.5)$$

$\hat{x}(k|k-1)$ time updated state vector

$\hat{x}(k|k)$ measurement updated state vector

$\mathbf{p}(k|k-1)$ time updated error covariance

$\mathbf{p}(k|k)$ measurement updated error covariance

Φ state transition matrix

K Kalman gain matrix

I identity matrix

H measurement transformation matrix

R measurement error covariance matrix

w process noise

Q process noise covariance matrix

$\upsilon(k) = (z(k) - H\,\hat{x}(k|k-1))$ measurement residual

$\Psi(k) = H\,\mathbf{p}(k|k-1)\,H^{T} + R$ measurement residual covariance

Once the initial state $\hat{x}(0|0)$ and the initial error covariance matrix $\mathbf{p}(0|0)$ are established, the Kalman equations can be activated. Figure 5.2, shows a simplified scheme of Kalman filter operation.

When Kalman introduced these equations ((5.1) - (5.5)) 27 years ago [Ref. 29], they offered the engineering community a means to model discrete - time systems via the state-space modeling method for multivariable systems. Unfortunately, when the state equation was written for the tracking applications, Kalman's conventional equations were found to fall short in two respects:

a. Finite wordlength computation.

b. Changing system dynamics model in real time.

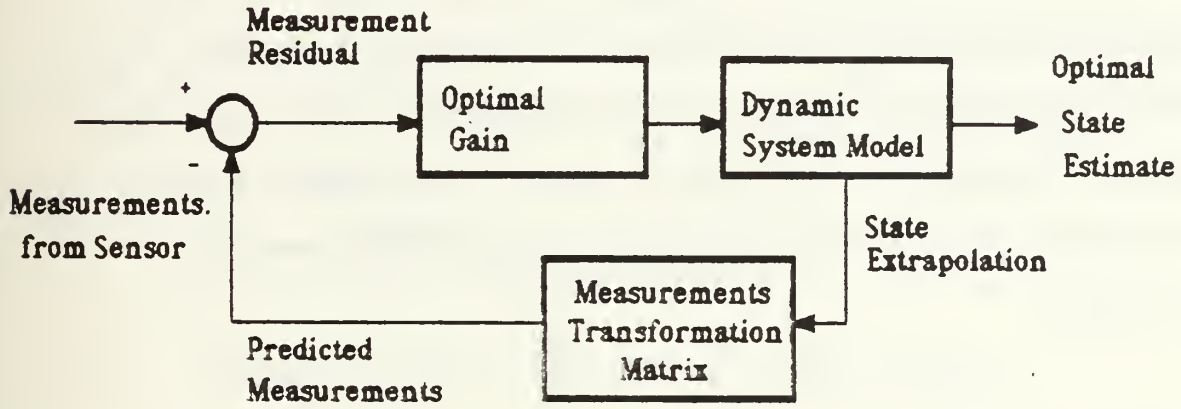


Figure 5.2. Simplified Scheme Of Kalman Filter

C. THE EXTENDED KALMAN FILTER

For nonlinear modeling problems, the Extended Kalman Filter (EKF) is used to extend the linearized Kalman filter design by relinearizing about each estimate (i.e., Kalman equation (5.1)) once it has been computed. The sucess of the method of linearization about a nominal trajectory in state space depends upon the accuracy of the nominal trajectory. This technique has little hope of success in a situation where there is almost no prior information about the trajectory as in the situation of the target acquisition problem. Here, a psuedo a priori may be generated from the incoming observations. In EKF, as soon as a new state estimate is made, a new and better reference state trajectory is incorporated into the estimation process. In this manner, one enhances the validity of the assumption that deviations from the reference (nominal) trajectory are small enough to allow linear perturbation techniques to be employed with adequate results [Ref. 27]. Radar measurements (range, azimuth, elevation, and possibly range rate) are in polar (spherical) coordinates. Thus there exist a known, nonlinear relationship or transformation between the state of the system and observations.

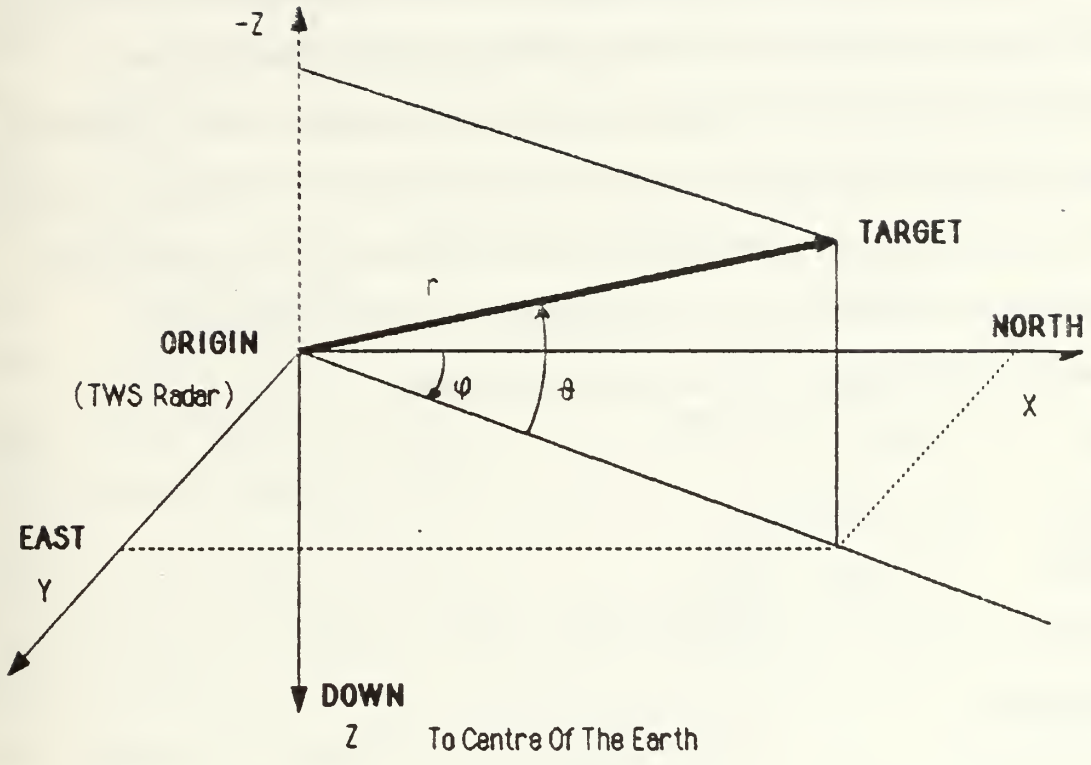


Figure 5.3. NED Coordinate Frame for TWS System

Assuming NED coordinate frame shown in Figure 5.3, where X,Y,and Z are North, East, and Down, respectively. The NED coordinate frame is chosen because it is applicable for surface (ground or shipbased) tracking systems, in addition that it is particularly useful for airborne systems. As shown in Figure 5.3, the origin of an aircraft tracking system is the TWS system site. It is worth to note that the NED system is not strictly an inertial system for a moving platform because the platform axes are slowly changing their orientation in space as the vehicle moves over the earth's surface. However, except near the North pole the effects of the rotations are negligible, and the NED system is essentially inertial for aircraft platform. For the coordinate system shown in Figure 5.3, we will have the following relationships between spherical (polar) and cartisian coordinates

Range $r = (x^2 + y^2 + z^2)^{0.5}$ (5.6)

Azimuth $\varphi = \arctan(y/x)$ (5.7)

Elevation $\vartheta = \arcsin(-z/r)$ (5.8)

Range rate $\dot{r} = (x\dot{x} + y\dot{y} + z\dot{z})/r$ (5.9)

Denoting that

$$x_e = \hat{x}(k|k-1) \quad , \quad \dot{x}_e = \hat{\dot{x}}(k|k-1)$$
$$y_e = \hat{y}(k|k-1) \quad , \quad \dot{y}_e = \hat{\dot{y}}(k|k-1) \qquad (5.10)$$
$$z_e = \hat{z}(k|k-1) \quad , \quad \dot{z}_e = \hat{\dot{z}}(k|k-1)$$
$$r_e = (x_e^2 + y_e^2 + z_e^2)^{0.5}$$

The corresponding measurement transformation matrix H, is given by [Ref. 28]

$$
H = \begin{bmatrix}
h_{11} & 0 & h_{13} & 0 & h_{15} & 0 \\
h_{21} & 0 & h_{23} & 0 & 0 & 0 \\
h_{31} & 0 & h_{33} & 0 & h_{35} & 0 \\
h_{41} & h_{42} & h_{43} & h_{44} & h_{45} & h_{46}
\end{bmatrix}
\begin{matrix}
\text{) Range} \\
\text{) Azimuth} \\
\text{) Elevation} \\
\text{) Range Rate}
\end{matrix}
\qquad (5.11)
$$

4 Measurements ↓ (rows)

→ 6 States (columns)

Where

$$
\begin{aligned}
h_{11} &= x_e / r_e \\
h_{13} &= y_e / r_e \\
h_{15} &= z_e / r_e \\
h_{21} &= - y_e / (x_e^2 + y_e^2) \\
h_{23} &= x_e / (x_e^2 + y_e^2) \\
h_{31} &= z_e x_e / ((r_e^2)(x_e^2 + y_e^2)^{0.5}) \\
h_{33} &= y_e z_e / ((r_e^2)(x_e^2 + y_e^2)^{0.5}) \\
h_{35} &= (- (x_e^2 + y_e^2)^{0.5}) / (r_e^2) \qquad (5.12) \\
h_{41} &= (\dot{x}_e / r_e) - ((x_e(x_e \dot{x}_e + y_e \dot{y}_e + z_e \dot{z}_e)) / r_e^3) \\
h_{42} &= x_e / r_e \\
h_{43} &= (\dot{y}_e / r_e) - ((y_e(x_e \dot{x}_e + y_e \dot{y}_e + z_e \dot{z}_e)) / r_e^3) \\
h_{44} &= y_e / r_e \\
h_{45} &= (\dot{z}_e / r_e) - ((z_e(x_e \dot{x}_e + y_e \dot{y}_e + z_e \dot{z}_e)) / r_e^3) \\
h_{46} &= z_e / r_e
\end{aligned}
$$

D. UD COVARIANCE FACTORIZED KALMAN FILTER

The most troublesome numerical aspect of the Kalman filter is the measurement update of the error covariance matrix, so the properties of alternate computational forms are of substantial interest. Potter [Ref. 30]

introduced a square-root approach for propagating the error covariance matrix in the absence of process noise. This method is completely successful in maintaining the positive semidefinite nature of the error covariance, and it effectively reduced the precision requirements to about half the number of bits needed for the full covariance update. The outstanding numerical characteristics and relative simplicity of this Potter square-root approach led to its implementation in the Apollo navigation filters [Ref. 31]. The numerical characteristics of the filter have been further improved upon by Agee and Turner [Ref. 32], Carlson [Ref. 33] and Bierman [Ref. 34], among others. The benefits of square-root filters were attractive in the sense of relieving the numerical problems and maintaining symmetry of the covariance matrix. Bierman's method requires the fewest arithmetic operations of the square-root formulations. Bierman's UD filter factorizes the covariance matrix P in the filter as follows

$$\mathbf{P} = UDU^T \tag{5.13}$$

where U is an upper triangular matrix with 1's along its main diagonal and D is a diagonal matrix. The UD algorithm requires a diagonal measurement covariance matrix R, which implies that the measurement errors are uncorrelated. If the matrix is not diagonal, it can be made diagonal as indicated by Yannone [Ref. 35]. Bierman derived a recursive algorithm for implementing a Kalman filter in terms of U and D, rather than $\mathbf{p}$ [Ref. 34, 36].

E. PARALLEL KALMAN FILTERING

In general, real-time filtering cannot be performed on large-scale problems using a uniprocessor architecture because serious processing lags can result. The Kalman filter can be extended to a much greater class of problems by using parallel processing concepts. Full utilization of parallelism can be obtained through insight in the structure of the problem and decoupling of arithmetic processes to permit concurrent processing. One must simultaneously develop the parallel algorithms for solving the filtering problem, and the associated processor architectures to achieve the maximum benefits from parallelism. Parallel Kalman filter architectures based on this design methodology can be implemented with VLSI/VHSIC technology [Ref. 37]. VLSI technology allows the designer to map system level architectures directly into hardware. To date, relatively little research has been conducted on restructuring the Kalman filter for parallel processing. Three approaches that have been considered include [Ref. 37]:

a. Vectorizing the standard Kalman filtering equations by running the filter on a vector (or array) processor [Ref. 38].

b. Uncorrelating the measurement data to the filter so that each measurement can be pipelined into each processor simultaneously [Ref. 39].

c. Decoupling the predictor and corrector equations in the filter so that these computations can be evaluated simultaneously on separate processors using multiprocessing [Ref. 36, 40].

Although the first approach can speed up computations considerably over conventional techniques (speed-up factors of 6 to 10 have been realised [Ref. 38]), the computational throughput was limited primarily by the architecture of the array processor. This occurred because the array

processor architecture was optimized for Fast Fourier Transform (FFT) computations, not Kalman filter computations.

An approach based upon mapping the Kalman filter equations directly onto a highly parallel/pipelined architecture can be used. Thus, the parallel Kalman filters algorithms/architectures developed in [Ref. 36] exploit the structure of the filter to improve throughput using pipelining and multiprocessing. This approach to speed up Kalman filter computations is to perform parallel processing at three major levels:

1. The measurement data to the filter is uncorrelated so that each measurement can be processed simultaneously.
2. The predictor and corrector equations of the Kalman filter are decoupled so that the predictor and corrector can be computed on separate processors.
3. The measurement data are pipelined into each processor.

Thus, multiprocessing and pipelining are combined to achieve large improvements in computational speed. Each processor architecture can be implemented with VLSI technology. To facilitate parallel processing and pipelining, the measurement data to the filter should be uncorrelated. The data can be uncorrelated by diagonalizing the covariance matrix associated with the measurement noise in the filter. Procedures for uncorrelating the data generally utilize coordinate transformations based upon eigenvalue or singular value decomposition [Ref. 41]. If the eigenvalues of R are not distinct, the generalized eigenvectors of R must be determined [Ref. 41]. Similarly, if the eigenvalues of R are complex, R can be transformed to a block diagonal matrix [Ref. 41].

F. ROBUSTNESS OF THE KALMAN FILTER

An important part of the design of the Kalman filter is to regulate the robustness of the filter to handle deviations from the prescribed systems dynamics model in the cases when the target makes heading changes. Without some form of re-modeling adaptively in real-time, the filter diverges. When the conventional filter is emulated for constant velocity/constant heading targets, filter divergence results after a number of iterations, and the error covariance matrix goes negative definite. This lack of reliability forces the seeking of a more stable algorithm for implementing the Kalman filter. The estimator eigenvalues control [Ref. 28], can be used. In a Kalman filter, this amounts to the eigenvalues of $(\Phi - KH)$. The control for any given (Φ, H) pair, is a function of K , K is a function of $\mathbf{p}(k+1|k)$, and $\mathbf{p}(k+1|k)$ is a function ofQ.

Whatever method is adopted for track filtering, it is usually necessary to combine it with some form of adaptation. An adaptive system is one which continually adjusts its own parameters in the course of time to meet a certain performance criterion. On-line adaptation is required when significant changes occur in the target motion (maneuvers), measurement accuracy or frequency of detection. The excitation noise covariance Q is a statistical quantity used to cover uncertainties in the model of target motion described by (4.5). Measurement accuracy is described statistically by the noise covariance R, and the interval between filter updates is given by the time T (which appears in Φ,Q(k)). Changes in the tracking environment must be reflected in the appropriate adjustment of these three filter parameters during the track estimation process. Adaptive tracking requires

the on-line computation of a figure of merit , or track performance indicator, which typically involves a weighted combination of terms in the residual (the innovation sequence) $\nu(k)$. It also requires a practical procedure for determining what quantitative adjustments should be made in the filter parameters. Two approaches can be used to control the filter performance:

a. The system's eigenvalues are fixed and kept constant. By holding the eigenvalues constant, a prescribed degree of stability is maintained.

b. A threshoiding technique is mechanized to adaptively model maneuvering/nonmaneuvering targets.

The first method, which computes the eigenvalues as function of the Q-matrix with constraints based upon the damping sought (critical, overdamped), starts with Kalman equations (5.2), (5.3), and (5.4):

$$K = \text{function of } (\mathbf{p}(k+1|k), R, H)$$

$$\mathbf{p}(k+1|k) = \text{function of } (\mathbf{p}(k|k), Q, \Phi)$$

Solving generally, for $\mathbf{p}(k+1|k)$ and substituting into the equation for K leads to the stability:

$$(\Phi - KH) = \text{function of } (\mathbf{p}(k|k), Q, \Phi, R, H)$$

The eigenvalues of $(\Phi - KH)$ can be sought, to yield a characteristic polynomial

$$q(\xi) = \xi^n + a_1 \xi^{n-1} + a_2 \xi^{n-2} + \cdots\cdots\cdots + a_{n-1} \xi + a_n \qquad (5.14)$$

For n = 6, (5.14) will be

$$q(\xi) = \xi^6 + a_1 \xi^5 + a_2 \xi^4 + a_3 \xi^3 + a_4 \xi^2 + a_5 \xi + a_6 \qquad (5.15)$$

Values of ξ_1 through ξ_5 can be chosen to yield a prescribed system response in the z-plane. The unit circle represents underdamping (marginally stable system, poles on the jw-axis in the s-plane). Within the unit circle represents either critical or overdamped system responses. Selecting critical

damping leads to a polynomial $q(\xi)$. The coefficients of the polynomial are functions of ($\mathbf{p}(k|k)$, Q, Φ, R), from which equations of all non-zero Q- matrix elements can be computed as functions of $\mathbf{p}(k|k)$, Φ, R, and constants. Thus constrained eigenvalues of (Φ - KH) imply a prescribed system response (critical, underdamped, or overdamped). They are regulated by resulting expressions of variable Q- matrix elements, which are, themselves, a function of $\mathbf{p}(k|k)$, Φ, and R.

The problem with this is that the optimum estimates may not always be derivable from a priori, critically-damped eigenvalues. Keeping the filter bandwidth open (high) all the time is not the optimal solution for the multitarget tracking problem. As a consequence, the other method is more adequate.

The second method is a thresholding mechanism to adaptively open and close(i.e., raise and lower) the filter bandwidth based upon information about the target's actual activity. Trial and error led to a "high" and "low" Q based on the range and range rate residuals. Emulation of the system TWS estimator revealed that a system response lag of one frame could be accommodated by evaluating the residuals prior to evaluating Kalman equation (5.2). If the threshold was broken, the Q- matrix was set to "high" or "low" accordingly, then equation (5.2) was evaluated with this selscted Q. This effectively gave the filter a "sneak preview" one T ahead with which to optimize the system robustness/stability. The fact that the system dynamics mode does change is the reason why the first method is not recommended to be used. The Kalman filter stability/robustness control can be summarized as the following:

$$\xi(\Phi - KH) < 1 \Rightarrow \text{stable}$$
$$\xi(\Phi - KH) = 1 \Rightarrow \text{marginally stable} \quad (5.16)$$
$$\xi(\Phi - KH) > 1 \Rightarrow \text{unstable}$$

a. Transient response varies with the closed loop filter pole locations.

b. The input to the Kalman filter which gives an indication that the target is maneuvering, are the residuals.

c. Residuals regulate Q, and Q regulates Kalman filter bandwidth, robustness, and stability.

When a high Q is used on a non-maneuvering target, the filter is being erroneously told that the constant velocity/constant heading system dynamics model is being violated when in fact it isn't, and vice versa. Conseqently, the filter estimates are noisier (less accurate) for non-maneuvering/"high" Q than necessary, and conversely, the filter diverges (breaks track) for the maneuvering/"low" Q case. In addition to the adaptive thresholding techniques, the state vector estimate accuracy was improved by performing a relinearization about the best estimate sequentially for each measurement made. Figure 5.4, shows the sequence of computations.

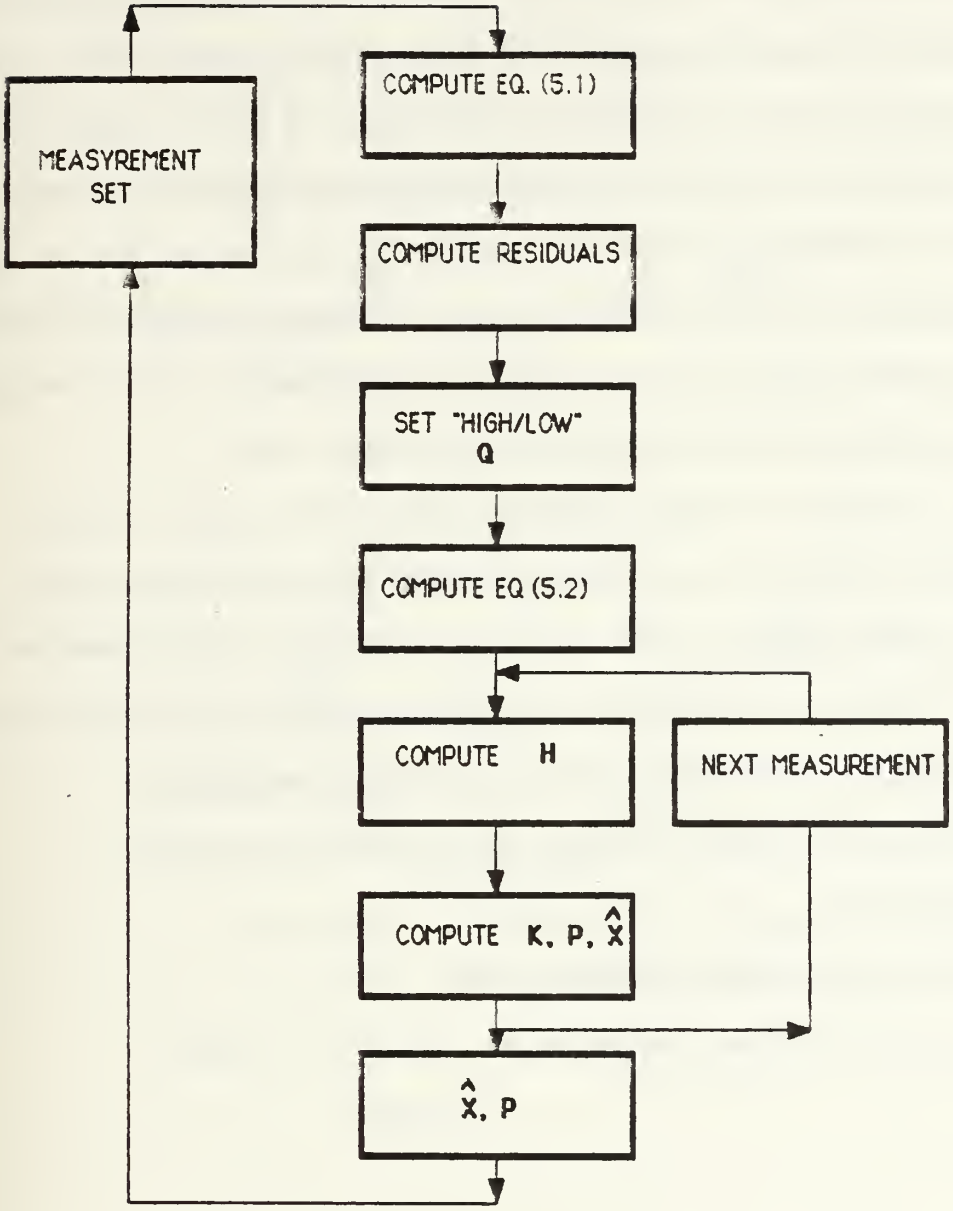


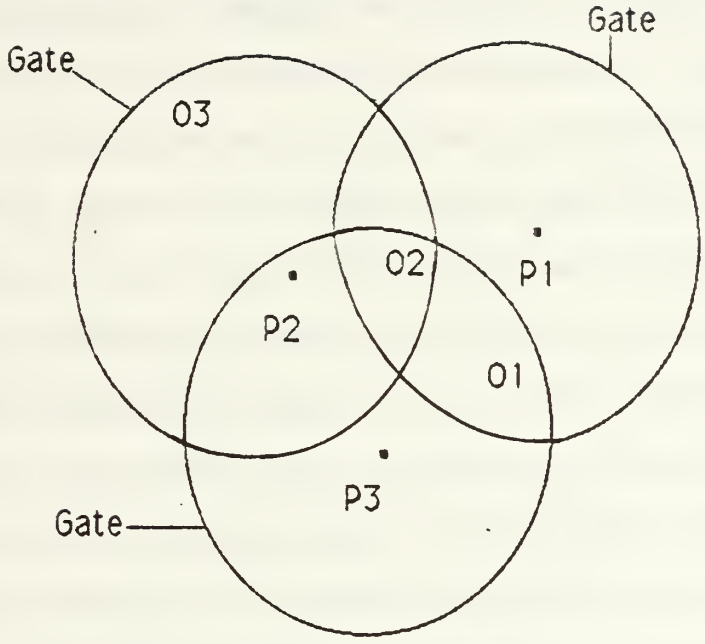
FIGURE 5.4. Computation Sequence and Sequential Relinearization About The Best Estimate

G. MULTITARGET LOCAL ESTIMATORS

It was not until the early 1970's, that multitarget tracking theory became a major topic of interest [Ref. 15, 18]. The papers by Singer and Stein [Ref. 42], Singer, Sea and Housewright [Ref. 43], Jaffer and Bar-shalom [Ref. 44], Bar-shalom and Tse [Ref. 45], began the development of modern multitarget tracking techniques that combine correlation and Kalman filtering theory [Ref. 15, 18]. In a dense target environment, gating only begins to solve the problem of associating observations with tracks. Additional logic is required when an observation falls within the gates of multiple target tracks or when multiple observations fall within the gates of a target track.

Figure 5.5, illustrates a typical situation in which both types of conflict occur. This logic is required to face the ambiguity about the origin of the observations, which originated from the target of interest. The correlation function takes the output of the gating function and makes final observation-to-track assignments. The various measures proposed against the correlation conflict in automatic tracking can be reduced to the two substantial principles :

1. Application of the nearest- neighbor- rule.
2. Application of branching procedures.



O1, O2, O3 Observation Positions

P1, P2, P3 Predicted Target Positions

Figure 5.5. Example of A Complex Conflict Situation

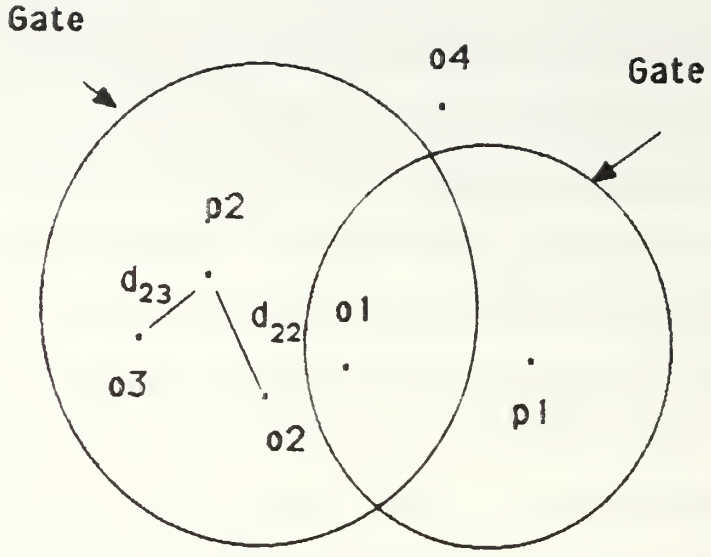
1. Nearest-Neighbor Approach

The nearest-neighbor-rule prescribes correlation of that track-result which is statistically nearest to the target's predicted position (the search plot in case of track initiation). It is based on the assumption, that the return which is nearest to the predicted position, is most probably the true target's return. The nearest-neighbor-rule is capable of solving satisfactorily the correlation conflict. At smaller target distances it fails to solve the conflict, because miscorrelations generally produce deviations of the track and target trajectory and finally lead to target loss. The track updating process typically begins with a gating procedure that is used to eliminate unlikely observation-to-track pairings. The simplest multitarget system use sequential data processing and the nearest-neighbor association rule. For example, this is normally the approach used with a TWS system. With this approach, processing is done at each scan using only data received on that scan to update the results of previous processing. The nearest-neighbor assignment algorithm assigns observations for existing tracks in a manner that minimizes some overall distance criterion. It looks for a unique pairing so that at most one observation can be used to update a given track. Using this approach, the optimal solution is obtained by assigning observations to tracks in order to minimize the total summed distance from all observations to the tracks to which they are assigned. A computationally efficient suboptimal solutions, also can be used to illustrate one suboptimal solution. To illustrate one suboptimal solution, the example shown in Figure 5.6, is solved using the following rules:

1. O1 is assigned to T1 because O1 is the only observation within the gates of T1 while T2 has other observations (O2, O3) within its gates.

2. O3 is assigned to T2 because O3 is closer than O2 ($d_{23}^2 < d_{22}^2$).

3. O4 can, without question, be used to initiate a new track, but new track initiation using O2 may be restricted. This restriction is based upon the practical consideration that multiple observations within the gate of a single established track are often the result of a failure in the observation redundancy-elimination logic. Thus, this restriction serves to prevent initiation of extraneous tracks.



o1, o2 o3, o4 = Observation Positions

p1, p2 = Predicted Target Positions

d = Distance From p2 to o2

d = Distance From p2 to o3

Figure 5.6. Example of Gating and correlation for Two Closely Spaced Tracks

Simple assignment techniques, such as the sequential nearest-neighbor approach, can lead to miscorrelation with poor tracking as a consequence. The problem with choosing the nearest-neighbor is that, with some probability, it is not the correct measurement. Therefore, it will use (sometimes) incorrect measurements while believing that they are correct. This can lead to the loss of target.

An alternative to nearest-neighbor correlation is the "all neighbor" approach, which incorporates all observations within the neighborhood, as defined by the gate around the predicted target position. The position update is then based on a weighted sum of all observations, with the weighting calculated using probability theory [Ref. 45]. This procedure is called Probabilistic Data Association (PDA) since it associates probabilistically all the neighbors to the target of interest. Then this probabilistic information is used in a suitably modified tracking filter, called PDA Filter (PDAF), That accounts for the measurements origin uncertainty. Later results [Ref. 21,46] showed that the PDA did not perform well in the presence of multiple targets, so a modified method denoted Joint Probabilistic Data Association (JPDA) was derived to include the presence of multiple targets [Ref. 21].

2. Branching Procedures

Branching procedures use all the results within the correlation gate in order to form new tracks. So, at every sampling time when there is more than one measurement in the validation region (correlation gate) the track is split. The likelihood function of each split track is evaluated in order to eliminate unlikely tracks. Tracks whose likelihood is below a given threshold are disregarded so as to keep the number of branches finite. The

likelihood of a measurement z(k), under a given assumption Ω_i for the model is obtained from z(k) and the predicted state $\hat{x}(k|k-1)$ as

$$p(z(k)|\Omega_i) = C \exp\{-1/2\ v^T(k)\Psi^{-1}(k)v(k)\} \quad (5.17)$$

When c is a normalizing constant. In the track- splitting techniques, it is necessary to evaluate the likelihood of a whole track, i.e. of a succession of plots. The fundamental limitation of the maximum likelihood techniques is that no validation test is made to control the truth of a given assumtion. This in practice leads to a feedforward approach to adaptivity by comparison with the feedback concept underlying the Bayesian approach. The branching procedures are marked by an unavoidable increase in the number of branch-tracks in comparison to the number of real targets. With regard to the application in a real-time system, these methods are generally computationally costly, so they are often inappropriate as true supplement to the nearest-neighbor approach.

Involving large computer burden and extensive memory requirements, sub-optimal techniques are generally preferred, leading to simplified algorithms and requiring storage of less data. In practice, a trade-off between cost and effectiveness is needed to select the most appropriate algorithm for each application. In HEA approach, the output of the local estimator is assumed to be tracks which are formed and confirmed with great confidence and low degree of uncetainty.

VI. ALGORITHMIC TRACK FUSION

A. TRACK-TRACK ASSOCIATION

Let $\hat{x}^{ii}$ be the local state estimate of a target i by a sensor i and $\hat{x}^{jj}$ be the local state estimate of a target j by a sensor j. Local estimates $\hat{x}^{ii}$ and $\hat{x}^{jj}$ could be for the same target or could be for different targets. It is desired first to test the hypothesis that these estimates pertain to the same target, in which case the two tracks will be associated as having a common origin. The optimal test would require the entire data base through time k and is probably not practical. In view of this, the test is carried out based only on the most recent estimates.

The decision as to whether track "i" provided by node "i" and track "j" provided by node "j", should be associated can be posed as a test of hypotheses by defining a distance between the two tracks, so

$$d^{ij}(i,j) = \| \hat{x}^{ii} - \hat{x}^{jj} \|^2 \qquad (6.1)$$

where $\| \hat{x}^{ii} - \hat{x}^{jj} \|$ is the norm of $[\hat{x}^{ii} - \hat{x}^{jj}]$

$$\| \hat{x}^{ii} - \hat{x}^{jj} \| = (E\{[\hat{x}^{ii} - \hat{x}^{jj}]^T [\hat{x}^{ii} - \hat{x}^{jj}]\})^{0.5}$$

$$= E^{0.5}\{[\hat{x}^{ii} - \hat{x}^{jj}]^T [\hat{x}^{ii} - \hat{x}^{jj}]\}$$

$$= (\mathbf{tr}\, E\{[\hat{x}^{ii} - \hat{x}^{jj}]^T [\hat{x}^{ii} - \hat{x}^{jj}]\})^{0.5}$$

The distance between two vectors in the space is defined in the usual way from the norm of a vector , so

$$\| \hat{x}^{ii} - \hat{x}^{jj} \|^2 = E\{[\hat{x}^{ii} - \hat{x}^{jj}]^T [\hat{x}^{ii} - \hat{x}^{jj}]\} \qquad (6.2)$$

Which is assumed to have a Gaussian distribution. The superscripts i and j denotes the tracks i and j and (i,j) denotes the nodes i and j respectively. The statistical test is

$$r = [\, d^{i\,j}(i,j)\,]^{T} P^{-1} [\, d^{i\,j}(i,j)\,] \underset{H_0}{\overset{H_1}{\lessgtr}} a^2 \qquad (6.3)$$

Where hypothesis H_0: track i and track j belong to different targets

hypothesis H_1 : track i and track j belong to the same target

and a^2 can be chosen based on that r will have a chi-square distribution with the number of degrees of freedom equal to the number of elements in the state vector. The covariance matrix for the statistical distance P and the resulting fused track can be given using the techniques mentioned in [Ref. 47].

The test to accept or reject the hypothesis that the two tracks are from the same target is defined using the similarity threshold a^2

$r \leq a^2$, tracks are from the same target

$r > a^2$, tracks are from different targets

It is noted that in the H_1 hypothesis $E\{ d^{i\,j}(i,j) \} = 0$. The choice of a^2 will be based upon the chi-square properties of r with some experimentation probably required for the particular application. The value of a^2 would also probably be chosen as a function of the target density if known [Ref. 18]. Also a^2 can be chosen to achieve a specified probability of correct association P_C.

B. PROBABILITY OF CORRECT ASSOCIATION

Evaluation of the correct association probability P_C is now considered. The probability of false association P_F can be found in a similar way but with a cumbersome calculations, owing to the non-zero mean value of $d^{ij}(i,j)$. The probability P_C is equivalent to the probability that $d^{ij}(i,j)$ lies inside the hyperellipsoid in the n-dimensional space, defined by equation (6.3). It can be computed by resorting to an ortho-normalization procedure of the matrix **P**, which transforms the hyperellipsoid into an equivalent hypersphere. Hence,

$$P_C = 1/(2\pi)^{n/2} \int_0^a \exp(-r^2/2)\, p(r)\, dr \qquad (6.4)$$

Where $p(r)\,dr$ is the symmetric volume element in the in the n dimensional space. For n=1,2,3,4, this expression particularises to

For n=1

$$P_C = \sqrt{2/\pi} \int_0^a \exp(-r^2/2)\, dr = 2\epsilon(a) \qquad (6.5)$$

Where $\epsilon(.)$ is the error function defined as

$$\epsilon(a) = 1/\sqrt{2\pi} \int_0^a \exp(-x^2/2)\, dx \qquad (6.6)$$

For n=2

$$P_C = \int_0^a r \exp(-r^2/2)\, dr = 1 - \exp(-a^2/2) \qquad (6.7)$$

For n=3

$$P_C = \sqrt{2/\pi} \int_0^a r^2 \exp(-r^2/2)\, dr = 2\epsilon(a) - \sqrt{2/\pi}\, a \exp(-a^2/2) \quad (6.8)$$

For n=4

$$P_C = 0.5 \int_0^a r^3 \exp(-r^2/2)\, dr = 1 - (1 + a^2/2) \exp(-a^2/2) \quad (6.9)$$

As an example, the values of P_C for n = 1,2,3 and 4 are shown in the tables 6.1, 6.2, 6.3, and 6.4 respectively. It is noticed that from the results given in the previous mentioned tables that, for the same similarity threshold a, the probability of correct association decrease as the number of elements in the state vector increase. Generally, with a=3, a high probability of correct association is obtained. The results presented in tables 6.1, 6.2, 6.3, and 6.4 is produced by an innovative PC software product called TK!Solver (see Appendix B). TK!Solver shortcuts the problem-solving process by eliminating two steps of developing an algorithm and writing a program. Instead, the user puts the mathematical model expressing the relationships between the variables directly using standard algebraic notation. In other words, the user communicates with the computer at the level of relationships (represented by equations) rather than at the level of sequential programs and assignment statements. In it, the input and output information reside in a set of eight sheets and three subsheets that are viewed on the screen. The principal ones are the Rule Sheet, used for entering and displaying the equations or rules, and the Variable Sheet, used for displaying the variable names. The Variable Sheet also serves for

assigning input values and units as well as for displaying the results of the solution. As an example, Figures 6.2, 6.3, and 6.4 show the variable and rule sheets of equations 6.5, 6.7, and 6.9 respectively. Konopask and Jayaraman view TK!Solver by itself as an expert system primarily in the area of numerical problem solving (see Appendix B).

TABLE 6.1. PROBABILITY OF CORRECT ASSOCIATION FOR n=1

a	P_C
0.5	0.371773290
1.0	0.666557870
1.5	0.888640141
2.0	0.979560553
2.5	0.985390484
3.0	0.999999853
3.5	1.0
4.0	1.0
4.5	1.0
5.0	1.0
5.5	1.0
6.0	1.0
6.5	1.0
7.0	1.0
7.5	1.0
8.0	1.0

TABLE 6.2. PROBABILITY OF CORRECT ASSOCIATION FOR n=2

a	P_C
0.5	0.117503097
1.0	0.393469340
1.5	0.675347533
2.0	0.864664717
2.5	0.956063066
3.0	0.988891004
3.5	0.997812509
4.0	0.999664537
4.5	0.999959935
5.0	0.999996237
5.5	0.999999730
6.0	0.999999985
6.5	0.999999999
7.0	1.0
7.5	1.0
8.0	1.0

TABLE 6.3. PROBABILITY OF CORRECT ASSOCIATION FOR n=3

a	P_C
0.5	0.019708344
1.0	0.182616987
1.5	0.500087808
2.0	0.763596943
2.5	0.897749084
3.0	0.973408794
3.5	0.993891228
4.0	0.998929359
4.5	0.999856146
5.0	0.999985142
5.5	0.999998815
6.0	0.999999927
6.5	0.999999996
7.0	1.0
7.5	1.0
8.0	1.0

TABLE 6.4. PROBABILITY OF CORRECT ASSOCIATION FOR n=4

a	P_C
0.5	0.007190985
1.0	0.090204010
1.5	0.310113507
2.0	0.593994150
2.5	0.818760149
3.0	0.938900519
3.5	0.984414126
4.0	0.996980836
4.5	0.999554274
5.0	0.999949690
5.5	0.999995647
6.0	0.999999711
6.5	0.999999985
7.0	0.999999999
7.5	1.0
8.0	1.0

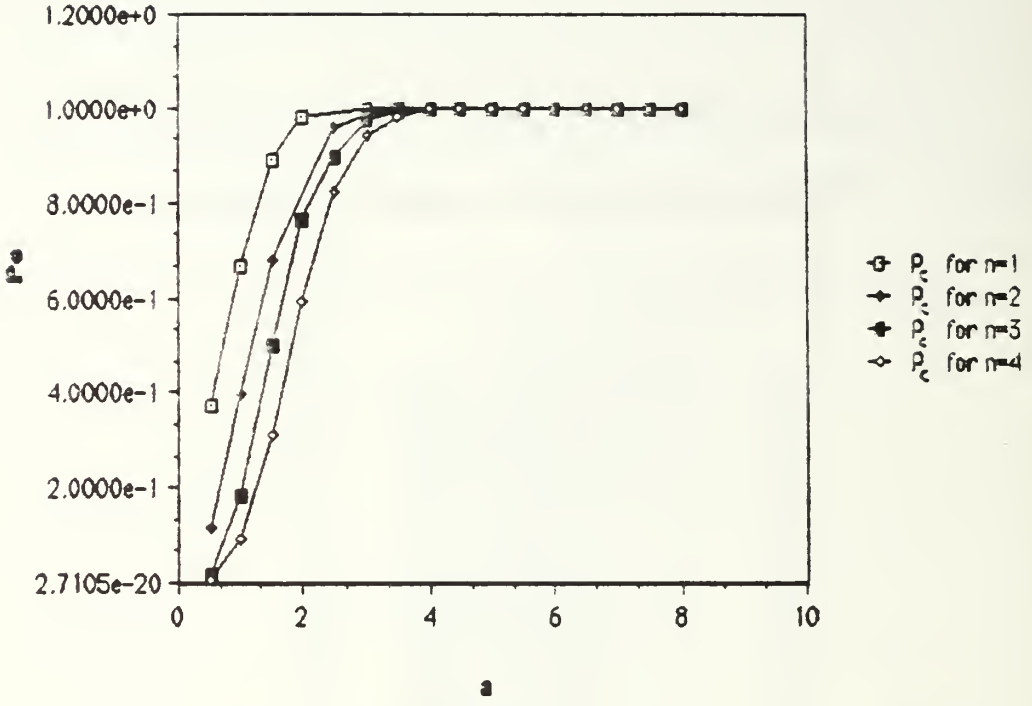


FIGURE 6.1. Probability of Correct Association P_C Versus Similarity Threshold a For Different Dimensional Space n of The State Vector

```
(7r) Rule:                                                        194/!

                    VARIABLE SHEET
St Input     Name    Output     Unit      Comment
-- -----     ----    ------     ----      -------
L            P       .88864014            Probability of correct association
L  0         a
L            f                            Function values
L  0         x                            Values of independent variable
L            S       1.1137466            Approximate value of integral
L  1.5       x2                           Upper limit
L  0         xl                           Lower limit
L  0         c                            Coefficients in simpson's formula
                    RULE SHEET
S Rule
- ----

* f=exp(-(x^2)/2)
* S=(x2-xl)/(3*10)*dot('c,'f)
* P=(2/sqrt(6.2832))*S
```

FIGURE 6.2. Variable and Rule Sheets For Equation (6.5)

```
(1i) Input: 0                                                    201/!

                    VARIABLE SHEET
St Input     Name    Output    Unit      Comment
-- -----     ----    ------    ----      -------
L  0         a
L            P
```

```
(1r) Rule: P=1-exp(-(a^2/2))                                     201/!

                    RULE SHEET
S Rule
- ----
* P=1-exp(-(a^2/2))
```

FIGURE 6.3. Variable and Rule Sheets for Equation (6.7)

```
(1i) Input:                                                        200/!

                    VARIABLE SHEET
St Input     Name    Output    Unit      Comment
-- -----     ----    ------    ----      -------
L            P
L  0         a


                    RULE SHEET
S Rule
- ----
* P=1-(1+(a^2/2))*exp(-(a^2/2))
```

FIGURE 6.4. Variable and Rule Sheet for Equation (6.9)

C. COMPOSITE ESTIMATE OF TWO TRACKS

Since the tracks are from different sensors and at the same time instant, the notation for sensors and time are dropped for simplicity of notation. If the tracks are independent, the covariance matrix P for d^{ij} is defined as

$$P = P^i + P^j \tag{6.10}$$

Bar-Shalom [Ref. 33], has pointed out that the covariance defined by (6.10) and the resultant formation of r are not strictly valid because of error correlation between the two sensor estimates. This correlation occurs, even if the measurement errors are independent, because of the common error source due to the target dynamics that are seen by both sensors. A technique outlined below, to account for this error correlation can be applied to modify the covariance matrix P.

Define a cross covariance matrix p^{ij} such that the initial condition is

$$p^{ij}(0|0) = 0$$

Then, for k> 0 values of $p^{ij}(k|k)$ are computed using the recursive relationship :

$$p^{ij}(k|k) = A^i(k)\, B(k-1)\, [A^j(k)]^T \tag{6.11}$$

Where

$$A^i(k) = I - K^i(k)\, h^i \tag{6.12}$$

$$A^j(k) = I - K^j(k)\, h^j \tag{6.13}$$

$$B(k-1) = \Phi^i\, p^{ij}(k-1|k-1)\, (\Phi^j)^T + Q(k-1) \tag{6.14}$$

The superscripts i and j refer to sensors i and j, while Φ,K,h, and Q are defined for the Kalman filter [Ref. 48,49]. Finally, the modefied covariance, replacing that given by (6.10) becomes [Ref. 47]

$$P = P^i + P^j - p^{ij} - [p^{ij}]^T \tag{6.15}$$

For combining tracks, we begin by considering the fusion relationships for a scalar such that a composite estimate, using estimates

$$\hat{x}^c = \hat{x}^1 + c(\hat{x}^2 - \hat{x}^1) \tag{6.16}$$

Where c is a weighting factor that will be chosen so that the expected Mean Squared Error (MSE) on $\hat{x}^c$ is minimized. The error in $\hat{x}^c$ is defined as

$$\Delta\hat{x}^c = \Delta\hat{x}^1 + c(\Delta\hat{x}^2 - \Delta\hat{x}^1)$$

Then, the error variance on Δx^c is defined as

$$e^2 = E[(\Delta\hat{x}^c)^2] = \sigma_1^2 + 2c\,E[\Delta\hat{x}^1\,\Delta\hat{x}^2] - 2c\,\sigma_2^2 + c^2\sigma_\Delta^2 \tag{6.17}$$

Where

$$\sigma_1^2 = E[(\Delta\hat{x}^1)^2], \qquad \sigma_2^2 = E[(\Delta\hat{x}^2)^2],$$

$$\sigma_\Delta^2 = E[(\Delta\hat{x}^2 - \Delta\hat{x}^1)^2] = \sigma_1^2 + \sigma_2^2 - 2\,E[\Delta\hat{x}^1\,\Delta\hat{x}^2]$$

The correlation between errors is defined as

$$E[\Delta\hat{x}^1\,\Delta\hat{x}^2] = R^{12}$$

Equation (22) becomes

$$e^2 = (1-2c+c^2)\sigma_1^2 + c^2\sigma_2^2 + 2(c-c^2)R^{12} \tag{6.18}$$

In order to form the minimum MSE estimates, we have

$$\partial e^2/\partial c = -2(1-c)\sigma_1^2 + 2c\,\sigma_2^2 + 2(1-2c)R^{12} = 0 \tag{6.19}$$

Solving (6.19) for c gives

$$c = (\sigma_1^2 - R^{12})/(\sigma_1^2 + \sigma_2^2 - 2R^{12}) \tag{6.20}$$

In the special case of no correlation

$R^{12} = 0$, we have

$$\sigma_1^2 = \hat{x}^1 + [\sigma_1^2/(\sigma_1^2 + \sigma_2^2)](\hat{x}^2 - \hat{x}^1)$$

$$= (\sigma_2^2 \hat{x}^1 + \sigma_1^2 \hat{x}^2) / (\sigma_1^2 + \sigma_2^2) \qquad (6.21)$$

In the case of combining state estimation vectors ($\hat{x}^1$, $\hat{x}^2$) , the same general relationships given by (6.16) and (6.17) are used, except that the variances become covariances:

$$\sigma_1^2 \rightarrow \mathbf{p}^1 \quad , \quad \sigma_2^2 \rightarrow \mathbf{p}^2$$

$$2R^{12} \rightarrow [\, \mathbf{p}^{12} + (\mathbf{p}^{12})^T \,]$$

and c becomes a weighting matrix:

$$\mathbf{c} = [\, \mathbf{p}^1 - \mathbf{p}^{12} \,] [\, \mathbf{p}^1 + \mathbf{p}^2 - \mathbf{p}^{12} - (\mathbf{p}^{12})^T \,]^{-1} \qquad (6.22)$$

Finally, using (23) and (25), the resulting error variances (or covariances) are

$$\sigma^2 (\hat{x}^c) = \sigma_1^2 - (\sigma_1^2 + R^{12})^2 / (\sigma_1^2 + \sigma_2^2 - 2R^{12}) \qquad (6.23)$$

In the special case of no correlation ($R^{12} = 0$), we have

$$\sigma^2 (\hat{x}^c) = \sigma_1^2 \sigma_2^2 / (\sigma_1^2 + \sigma_2^2) \qquad (6.24)$$

So, generally in the vector case, the resulting combined vector, which minimizes the expected error is

$$\hat{x}^c = \hat{x}^i + c (\hat{x}^j - \hat{x}^i) \qquad (6.25)$$

$$c = (\mathbf{p}^i - \mathbf{p}^{ij}) \mathbf{p}^{-1}$$

and the covariance matrix associated with the estimate of (6.25) is

$$\mathbf{p}^c = \mathbf{p}^i - (\mathbf{p}^i - \mathbf{p}^{ij}) \mathbf{p}^{-1} (\mathbf{p}^i - \mathbf{p}^{ij})^T \qquad (6.26)$$

As a simple example

Consider the two estimates, each a 2- dimensional vector, with the following covariance matrices

$$\mathbf{p}^1 = \begin{bmatrix} 10 & 5 \\ 5 & 10 \end{bmatrix}$$

$$\mathbf{p}^2 = \begin{bmatrix} 10 & -5 \\ -5 & 10 \end{bmatrix}$$

Figure 6.5. presents the 1σ ellipses corresponding to these matrices, namely

$$\hat{\mathbf{x}}^T (\mathbf{p}^i)^{-1} \hat{\mathbf{x}} = 1 \qquad i=1,2 \qquad (6.27)$$

and 1σ ellipse corresponding to the fused estimate whose covariance is

$$\mathbf{p} = \mathbf{p}^1 (\mathbf{p}^1 + \mathbf{p}^2)^{-1} \mathbf{p}^2 \qquad (6.28)$$

The reduction of the uncertainty are noticed: the ellipse corresponding to the fused estimate is strictly smaller than the intersection of the two ellipses prior to fusion

Figure 6.6. shows the ellipses of uncertainty corresponding to

$$\mathbf{p}^1 = \begin{bmatrix} 10 & 3 \\ 3 & 10 \end{bmatrix}$$

$$\mathbf{p}^2 = \begin{bmatrix} 10 & -6 \\ -6 & 10 \end{bmatrix}$$

and the resulting fused estimate according to (6.25).

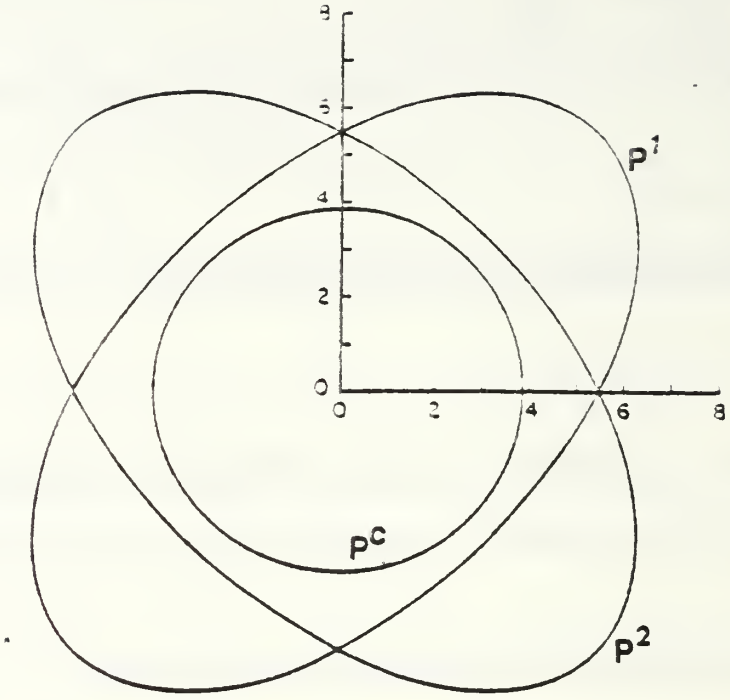


FIGURE 6.5. Error Ellipsoid For Fused Independent Tracks in Example 1

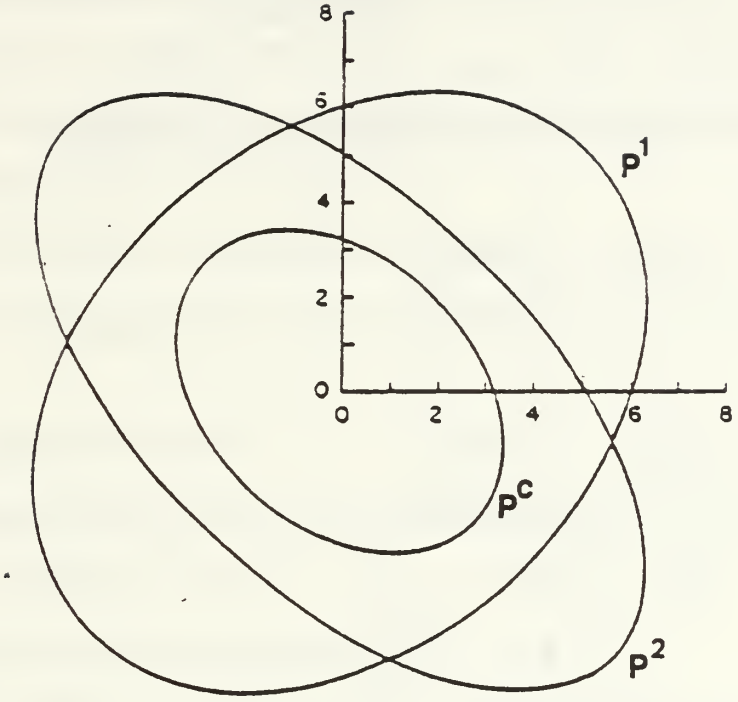


FIGURE 6.6. Error Ellipsoid For Fused Independent Tracks in Example 2

D. OPTIMALITY OF HEA TRACK FUSION

Assuming that there are two sites (nodes) at which the same target is tracked. At each site, each local estimates $\hat{x}^{i\,j}(k|k)$, with the corresponding $\mathbf{p}^{i\,j}(k|k)$ are computed. Hence, in this situation we are concerned only about one target, the superscript i is dropped for the simplicity of notations, and there will be $\hat{x}^{j}(k|k)$ with its corresponding $\mathbf{p}^{j}(k|k)$, with j=1,2.

Assuming the local measurements at each site are

$$z^{j}(k) = H^{j}(k)\, x(k) + v^{j}(k) \qquad j = 1,2 \tag{6.29}$$

with $R^{j}(k)$ the corresponding measurement noise covariance.

Denoting

$$z^{j}(k) = \begin{bmatrix} z^{1}(k) \\ z^{2}(k) \end{bmatrix} \tag{6.30}$$

$$H^{j}(k) = \begin{bmatrix} H^{1}(k) \\ H^{2}(k) \end{bmatrix} \tag{6.31}$$

$$R^{j}(k) = \begin{bmatrix} R^{1}(k) & \mathbf{0} \\ \mathbf{0} & R^{2}(k) \end{bmatrix} \tag{6.32}$$

Using the matrix inversion lemma

$$(\mathbf{p}^{-1} + H^{T} R^{-1} H)^{-1} = \mathbf{p} - \mathbf{p}\, H^{T} (H\, \mathbf{p}\, H^{T} + R)^{-1} H\, \mathbf{p} \tag{6.33}$$

which can also be written as

$$(\mathbf{p} + H\, R\, H^{T})^{-1} = \mathbf{p}^{-1} - \mathbf{p}^{-1} H (H^{T} \mathbf{p}^{-1} H + R^{-1})^{-1} H^{T} \mathbf{p}^{-1} \tag{6.34}$$

the recursion for the covariance $\mathbf{p}(k|k)$ can be rewritten as

$$\mathbf{p}(k|k) = [\mathbf{p}(k|k-1)^{-1} + H^{T}(k)\, R(k)^{-1} H(k)]^{-1} \tag{6.35}$$

$$= \mathbf{p}(k|k-1) - \mathbf{p}(k|k-1)\, H^{T}(k) + [H(k)\, \mathbf{p}(k|k-1) H^{T}(k) + R(k)]^{-1} H(k)\, \mathbf{p}(k|k-1) \tag{6.36}$$

Defining

$$\Psi(k) = H(k)\, \mathbf{p}(k|k-1) H^T(k) + R(k) \tag{6.37}$$

$$K(k) = \mathbf{p}(k|k-1)\, H^T(k)\, \Psi^{-1}(k) \tag{6.38}$$

the recursion for the covariance $\mathbf{p}(k|k)$ can be rewritten more compactly as

$$\mathbf{p}(k|k) = [\, I - K(k) H^T(k)]\, \mathbf{p}(k|k-1)]$$

$$= \mathbf{p}(k|k-1) - K(k)\Psi(k)H^T(k) \tag{6.39}$$

and the following identity can be written

$$\mathbf{p}(k|k)\, H^T(k)\, R(k)^{-1} = (\, \mathbf{p}(k|k-1)H^T(k) -$$

$$- \mathbf{p}(k|k-1)H^T(k)[H(k)\, \mathbf{p}(k|k-1)H^T(k) + R(k)]^{-1} H(k)\, \mathbf{p}(k|k-1)H^T(k))R(k)^{-1}$$

$$- \mathbf{p}(k|k-1)H^T(k)[H(k)\, \mathbf{p}(k|k-1)H^T(k) + R(k)]^{-1} *$$

$$(H(k)\, \mathbf{p}(k|k-1)H^T(k) + R(k) - H(k)\, \mathbf{p}(k|k-1)H^T(k))R(k)^{-1} = K(k)$$

So, the Kalman filter gain given by (5.3), can be given by the alternate expression

$$K(k) = \mathbf{p}(k|k)\, H^T(k)\, R(k)^{-1} \tag{6.40}$$

Which in the case of HEA, for each site (node) it will take the form

$$K^j(k) = \mathbf{p}^j(k|k)\, H^j(k)^T\, R(k)^{-1} \tag{6.41}$$

and equation (5.5) will have the form

$$\hat{x}^j(k|k) = \hat{x}^j(k|k-1) + K^j(k)\, (\, z^j(k) - H^j(k)\hat{x}^j(k|k-1)) \tag{6.42}$$

The recursion form for the inverse of the covariance update $\mathbf{p}^j(k|k)$ is given by the equation

$$\mathbf{p}^j(k|k)^{-1} = \mathbf{p}^j(k|k-1)^{-1} + H^j(k)^T\, R^j(k)^{-1} H^j(k) \tag{6.43}$$

The estimate $\hat{x}^j(k|k)$, using expression (6.41), (6.42) will be

$$\hat{x}^j(k|k) = \hat{x}^j(k|k-1) + \mathbf{p}^j(k|k)\, H^j(k)^T\, R^j(k)^{-1}\, (\, z^j(k) - H^j(k)\hat{x}^j(k|k-1)) \tag{6.44}$$

Multiplying (6.43) by (6.44) yields for $j=1,2$

$$\mathbf{p}^j(k|k)^{-1}\hat{x}^j(k|k) = [\mathbf{p}^j(k|k-1)^{-1} + H^j(k)^T\, R^j(k)^{-1} H^j(k)]\, \hat{x}^j(k|k-1)$$

$$+ \mathbf{p}^j(k|k)^{-1}\, \mathbf{p}^j(k|k)\, H^j(k)^T\, R(k)^{-1}\, (\, z^j(k) - H^j(k)\hat{x}^j(k|k-1))$$

$$- \mathbf{p}^{j}(k|k-1)^{-1}\,\hat{x}^{j}(k|k-1) + H^{j}(k)^{T}\,R(k)^{-1}\,z^{j}(k) \qquad (6.45)$$

Thus

$$H^{j}(k)^{T}\,R(k)^{-1}\,z^{j}(k) - \mathbf{p}^{j}(k|k)^{-1}\,\hat{x}^{j}(k|k) - \mathbf{p}^{j}(k|k-1)^{-1}\,\hat{x}^{j}(k|k-1)$$

(6.46) and this will be used to eliminate the measurements from the combined

estimation update equation which will be similar to (6.44) with the superscript c instead of j, i.e.

$$\hat{x}^{c}(k|k) - \hat{x}^{c}(k|k-1) + \mathbf{p}^{c}(k|k)\,H^{c}(k)^{T}\,R^{c}(k)^{-1}\,(\,z^{c}(k)-H^{c}(k)\hat{x}^{c}(k|k-1)) \qquad (6.47)$$

Using (6.30), (6.31) and taking advantage of the block-diagonal form of (6.32), the combined state updating, given by equation (6.47), can be rewritten as

$$\hat{x}^{c}(k|k) - \hat{x}^{c}(k|k-1) + \mathbf{p}^{c}(k|k)\sum_{j=1}^{2} H^{j}(k)^{T}\,R^{j}(k)^{-1}\,(\,z^{j}(k)-H^{j}(k)\hat{x}^{j}(k|k-1)) \qquad (6.48)$$

Similarly, the combined covariance update is similar to (6.41) with the superscript c instead of j, i.e.

$$\mathbf{p}^{c}(k|k)^{-1} - \mathbf{p}^{c}(k|k-1)^{-1} + H^{c}(k)^{T}\,R^{c}(k)^{-1}H^{c}(k) \qquad (6.49)$$

Similar to (6.48), (6.49) can be rewritten as

$$\mathbf{p}^{c}(k|k)^{-1} - \mathbf{p}^{c}(k|k-1)^{-1} + \sum_{j=1}^{2} H^{j}(k)^{T}\,R^{j}(k)^{-1}H^{j}(k) \qquad (6.50)$$

Multiplying (6.50) with (6.48) yields, after cancelations (similar to those in (6.45))

$$\mathbf{p}^{c}(k|k)^{-1}\hat{\mathbf{x}}^{c}(k|k) - \mathbf{p}^{c}(k|k-1)^{-1}\hat{\mathbf{x}}^{c}(k|k-1) + \sum_{j=1}^{2} H^{j}(k)^{T} R(k)^{-1} z^{j}(k) \qquad (6.51)$$

Finally, substituting (6.46) into (6.51), one obtains

$$\mathbf{p}^{c}(k|k)^{-1}\hat{\mathbf{x}}^{c}(k|k) - \mathbf{p}^{c}(k|k-1)^{-1}\hat{\mathbf{x}}^{c}(k|k-1)$$
$$+ \sum_{j=1}^{2} [\mathbf{p}^{j}(k|k)^{-1}\hat{\mathbf{x}}^{j}(k|k) - \mathbf{p}^{j}(k|k-1)^{-1}\hat{\mathbf{x}}^{j}(k|k-1)] \qquad (6.52)$$

Which is the sought-after expression of the combined estimate in terms of only the local estimates. So, the fused track estimates combine the local track estimates and the incoming track estimates without having to calculate cross covariance given by (6.11), and it can be stated that the fused estimates is the global estimates for the tracks in the overlapping area.

VII. KNOWLEDGE-BASED TRACK FUSION

A. CAPTURING THE EXPERTIZE OF A HUMAN EXPERT

In multitarget tracking we are concerned with the position of targets and their identity and behaviour. In fact, the position is of over-riding importance because identity and behaviour mean little unless they can be associated with position [Ref. 9]. Also, since we are concerned with a dynamic environment we need to take time into account. So, it would appear then that the first task with which we are faced is how to deal with kinematic information. In order to combine track information from any two radar sensors in a network, these are compared to determine whether they pertain to the same target. The decision process is called correlation. As we mentioned before, we consider this decision process implicitly included in the fusion process, because there is no meaning of the fusion without it. In real life, radar sensors provide different types of information with different accuracies. A modern radar display includes alphanumeric characters and symbols for directly conveying additional information. This is useful when target identity and altitude are to be displayed. The target track might be shown as a line on a synthetic display. The configuration of the line indicates the direction of the target path while its length can be made proportional to the target speed. This kind of display has a computer to generate the graphics and control the radar display. This permits magnification of a selected area, stored flight plans, stored clutter map and so on. The operator can communicate with the computer in an interactive

manner by means of a keyboard, track ball or light pen. Sometimes, an auxiliary display is mounted adjacent to the main display to provide tabular data that would otherwise encumber the main display [Ref. 15]. Current systems rely largly on manual correlations to form a coherent picture of the underlying situation. An operator at the radar display makes a visual comparison of the information provided by two or more sensors and by applying his experience arrives at a decision. But, even with moderately complex scenarios, the workload can easily overwhelm the limited number of operators available and displays. This can lead to important information being overlooked. A computer program which could reliably carry out a large fraction of this task would greatly assist the operation and performance of the entire system. The operators become proficient after a long time of practice. Discussions with former operators [Ref. 50], revealed that they acquired two things as they become experts. First, they memorized facts about the platforms operating in their area, the facilities available in the countries of the area and the political alliances of those countries. Secondly, they learned how to relate reports received to the above facts and the current situation in a way which would allow them to develop hypotheses about future platform motions and activities.

Our initial approach to the multisensor information fusion was to search for suitable techniques from the world of AI which:

1. Were well defined.
2. Had been demonstrated on a similar type of problem.

Perhaps the most well known and well defined part of the AI scene is expert systems.

With the arrival of expert systems , there is a great rush to encapsulate human expertise in computer software, and multisensor information fusion and its related fields are obvious areas for attention. The expert system can be considered in a wide sense as an application of artificial intelligence to computer software. The philosophy of an expert system is to produce a computer solution to a problem by capturing the expertize of a human expert. So, it can emulate the performance of a human expert by incorporating the analytic and heuristic knowledge which the human expert has. Of the many expert systems that have been developed in the last decade, the majority of the successful programs were designed to play a role analogous to that of a human consultant. Not surprisingly, this work was done in areas of Medicine, Chemistry, and Geology in which there is an established tradition for consultation. Human consultants in these fields are valuable because they are specialists who possess extensive knowledge about particular problem domains.

Expert consultation systems have focused on problems in which a human expert's knowledge is largly factual in nature. Here, the key to solving a problem lies more in knowing the relevant information than in ingeniously constructing a solution from logical principles. The human expert is distinguished by knowing all of the factors that are important, and by processing judgment in combining diverse considerations to reach a decision. It follows that a corresponding expert system must have effective ways to represent and employ large amounts of different kinds of knowledge bearing on specialized problems. Generaly, the expertise is in the form of rules, and these rules form the knowledge base of the system. Recently, expert

systems have been found effective in planning, monitoring, and interpretation tasks [Ref. 51]. Researches have applied this technology to a variety of military problems. For example, planning of aircraft missions [Ref. 52], simulation of air battles [Ref. 53], and analysis of platforms operating in a certain area, and their location and the activity in which they are engaged. The symbolic nature of the information fusion problem and absence of a well developed approach to it, suggested that a system with rapid prototyping capabilities would be helpful [Ref. 50]. Some expert system shells provide tools which make prototyping of data structure quick and the rules employed for reasoning easier to implement and modify than conventional programs. Figure 7.1, shows the general structure of an expert system.

B. KNOWLEDGE REPRESENTATION USING RULES

Rules provide a formal way of representing recommendations, directives, or strategies. They are often appropriate when the domain knowledge results from empirical associations developed through years of experience solving problems in an area. Rules are expressed as IF - THEN statements, as shown below.

IF : A is true
and B is true
and C is false
THEN : conclude X

This is a type of production rule which has the general form [Ref. 43]:

IF: logical conditions are satisfied
THEN: take the indicated action

When the IF portion of a rule is satisfied by the facts, the action specified by the THEN portion is performed. Among the potentially important assets of the production rules approach is that it provides the means of understanding how a decision was reached and is able to explain and correct erroneous conclusions.

A controlling framework is used to allow the user to access the knowledge base in the manner of a consultation whereby the user may volunteer information or the machine may question the user until sufficient evidence is gathered to produce useful conclusions. The user may also ask the system to explain its reasoning so that he may understand its reasoning and undestand how the conclusions were reached.

This method of problem solving was adopted as it seemed to fit the multisensor data fusion problem, assuming that the human expertise exists. However, most of the well-publicised expert systems are in quite different problem domains to multisensor information fusion. Examples being medical diagnosis, fault diagnosis and the well- known mineral prospecting expert system called PROSPECTOR. In this type of problem, it can be assumed that all symptoms belong to the same patient, whereas, in multisensor information fusion, there is the problem of finding out which evidence belongs to which patients, and indeed how many patients are present. Also the multisensor information fusion problem is a continuous, real-time problem, rather than a single shot diagnosis.

The heart of an expert system is its corpus of knowledge [Ref. 54]. When AI scientists use the term "knowledge", they mean the information a computer program needs before it can behave intelligently. The knowledge

in an expert system is organized in a way that separates the knowledge about the problem domain from the system's other knowledge, such as general knowledge about how to solve problems or knowledge about how to interact with the user. Several advantages accrue to this separation:

a. The same knowledge can be used for more than one purpose. For example, a given knowledge base can be used to solve a particular problem, to provide an explanation for the solution, or to support computer-aided instruction about the problem.

b. The power of the program can be extended either by expanding the knowledge base or by adding facilities to the interpreter. In particular, this allows a large system to be developed incrementally.

c. The problem-solving mechanisms and system facilities of the interpreter can be applied to similar problem domains by replacing the old knowledge base by a knowledge base for the new domain.

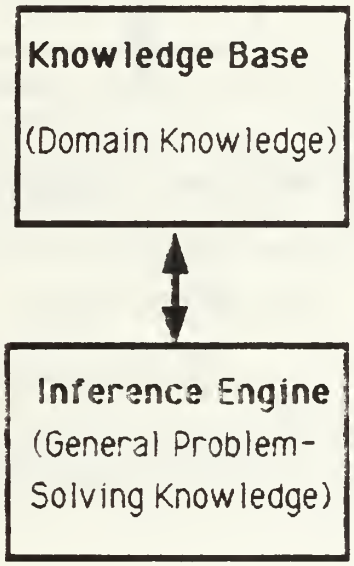


Figure 7.1. General Structure of an Expert System

The collection of domain knowledge is called the" knowledge base", while the general problem- solving knowledge is called the "inference engine". A software with knowledge organized this way is called a "knowledge-based system". Virtually, all expert systems are knowledge-based systems, while the converse is not necessarily true [Ref. 54].

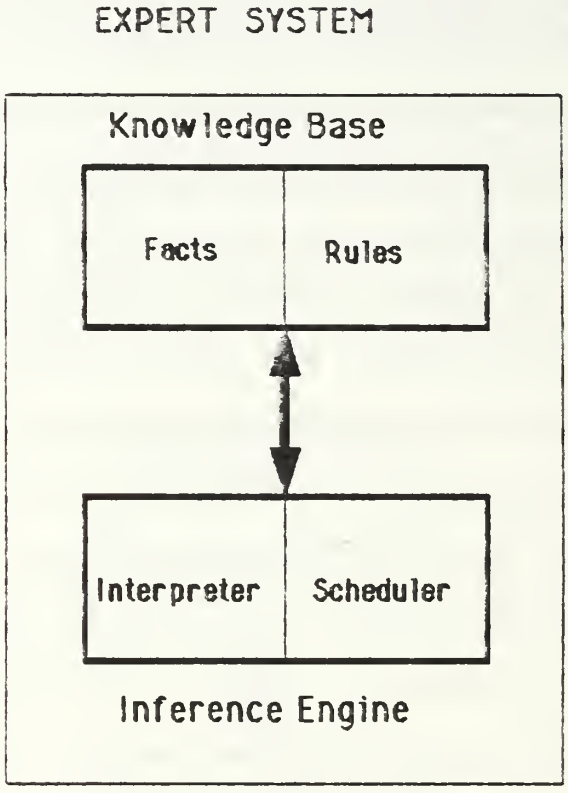


Figure 7.2. Structure of an Expert System

As shown in Figure 7.2, the knowledge base in an expert system contains facts (data) and rules that use those facts as the basis for decision making. The inference engine contains an interpreter that decides how to apply the rules to infer new knowledge and a scheduler that decides the order in which the rules should be applied. When the IF portion of a rule is satisfied

by the facts, the action specified by the **THEN** portion is performed. When this happens the rule is said to fire or execute [Ref. 55]. A rule interpreter compares the **IF** portions of rules with the facts and executes the rule whose **IF** portion matches the facts as shown in Figure 7.3.

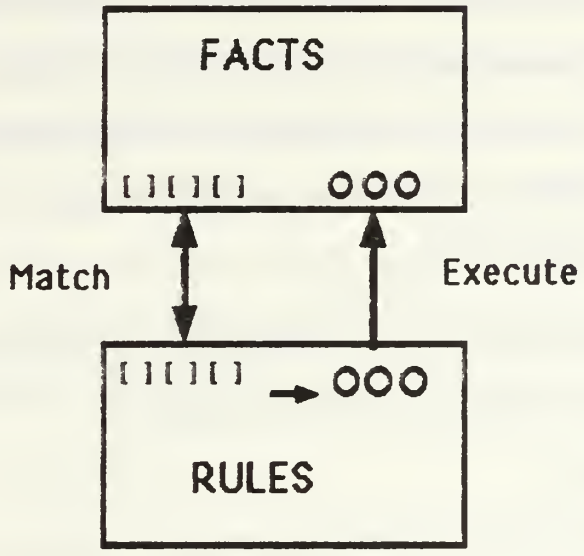


Figure 7.3. The Rule Interprets Cycles Through
a Match-Execute Sequence

C. INVOKING RULES IN A RULE-BASED SYSTEM

There are two important ways in which rules can be used and invoked in a rule-based system; one is called backward chaining and the other forward chaining.

1. Backward Chaining

Backward chaining is often described in terms of goal-directed reasoning or top-down reasoning. In backward chaining the system has a set of initial goals, and the rules are invoked in reverse order. The system

begins by examining a limited set of production rules, whose right-hand sides are the goals. The system then proceeds to examine the left-hand side of the rules to see which of the goals (RHS) are satisfied. As the rules are examined in this backward unraveling, some premises (of the left-hand side of rules) are unknown (logically unsatisfied) and therefore they become new subgoals. If a subgoal is unknown, a question may be asked to determine its status. Its strategy can be summarized in the following steps:

(1) Find a rule a "**THEN**" pattern that matches the goal.
    Found ------- Go to step 2.
    Not Found --- Fail.

(2) Use the "**IF**" part of the rule to establish new sub-goal(s).

(3) Find fact(s) that satisfies the new sub-goal(s).

2. Forward Chaining

In forward chaining the system does not start with any particular goals for it. That is, it has no initial subgroup of production rules which establish a starting point. Instead, the system starts with a subset of evidence and proceeds to invoke the production rules in a forward direction, continuing until no further production rules can be invoked. Its strategy can be summarized in the following steps:

(1) Find a rule with an "**IF**" pattern that matches a fact
    Found ----------Go to step 2.
    Not Found ------Fail to find a goal.

(2) Assert the rules "**THEN**" clause, i.e add a new fact to the data base.

(3) Does the new fact satisfy the goal?
    Yes ------------ we are successful, quit
    No -------------- Go to step 1.

3. Backward Versus Forward Chaining

Although several systems have been built emphasizing backward chaining, both forms of invocation and evaluation of the production rules are equally valid as long as they yield the same correct conclusions. The rate of arriving at the conclusions will probably differ considerably depending on the strategy adopted. Most classification problems can be solved using either one of the approaches individually or a mixture for production rule evaluation. The shape of the problem space determines which is better. As shown in Figure 7.4, fan-in calls for forward chaining and fan-out calls for backward chaining. The bidirectional search is often favorable. Using it the forward chaining begins from the known facts and the backward chaining begins from the best hypothesis.

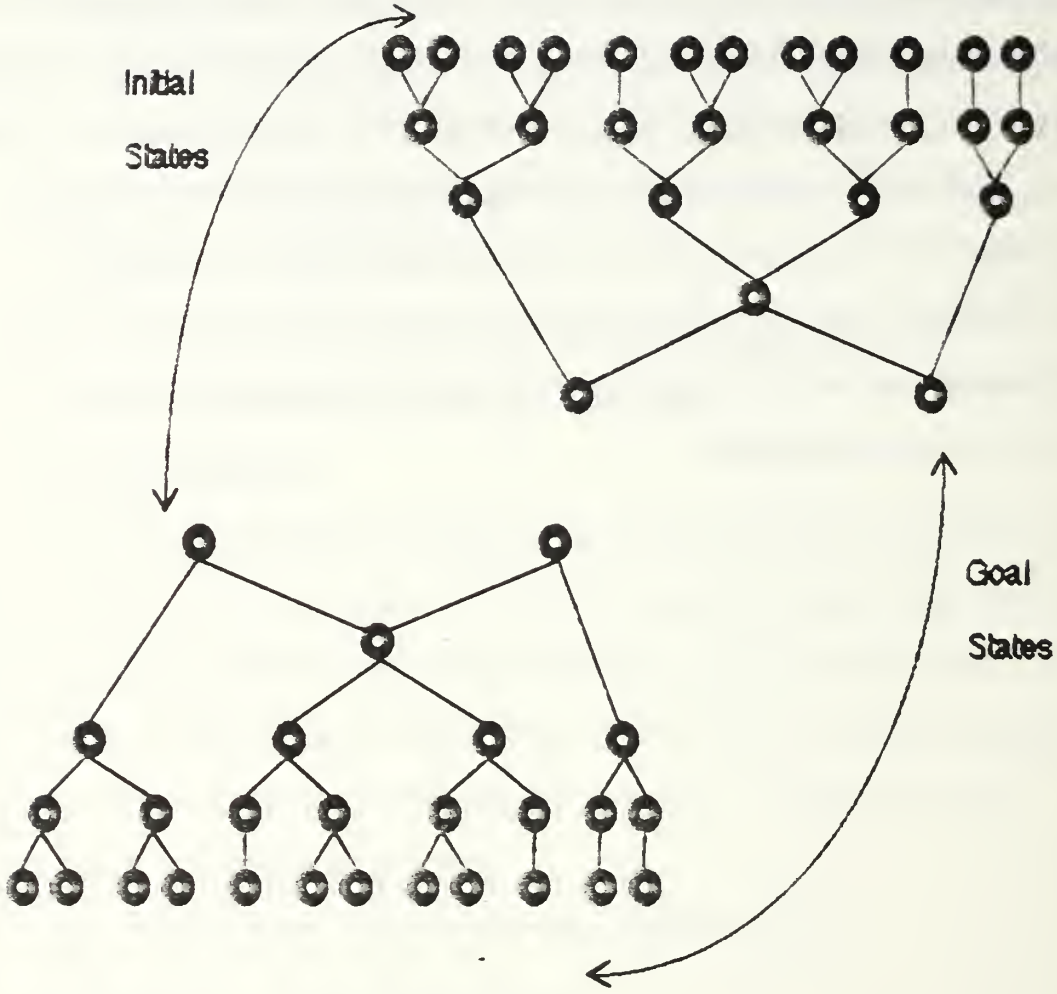


Figure 7.4. Fan-IN and Fan-Out Stages of Knowledge Aquisition

D. METAKNOWLEDGE AND EXPLANATION FACILITY

An expert system has knowledge that lets it reason about its own operations plus a structure that simplifies this reasoning process. This knowledge the system has about how it reasons is called "metaknowledge", which just means knowledge about knowledge. Also, it is better to have what is called an "explanation facility". This is knowledge for explaining how the system arrived at its answer.

E. BLACKBOARDS

Blackboards refers to a particular AI problem solving methodology. The best known applications of the blackboard methodology are HEARSAY-II, a speech understanding system [Ref. 56], and the HASP/SIAP sonar data interpretation system [Ref. 57, 58].These applications effectively processed regular streams of data from a single sensor, treating any other information as locally static. But the blackboard methodology is more generally applicable. In particular, it provides a convenient framework for integrating maximally reduced information from multiple sources with different temporal characteristics.

The blackboard problem solving methodology originated approximately 10 years ago and has been evolving ever since [Ref. 59]. The main feature of a blackboard system is a global data store holding input data and hypotheses about the solution of the problem derived from that data. Related information is kept together. The global data store is known as the blackboard.

F. REPRESENTING UNCERTAINTY IN EXPERT SYSTEMS

Expert systems are often forced to make judgements in the light of incomplete or unreliable data. The general problem of drawing inferences from uncertain or incomplete data has inspired a variety of technical approaches. Parade [Ref. 60] offers a review of different approximate reasoning techniques which have been proposed for dealing with uncertain or imprecise knowledge in expert systems. These tecniques can be summarized as Bayesian model, Dempster-Shafer belief theory, fuzzy logics, and ad hoc approaches.

1. Bayesian Model

One of the most useful results of probability theory is Bayes theorem, which provides a way of computing the probability of a particular event given some set of observation which is made. Let

$P(H_i|E)$ = the probability that hypothesis H_i is true given evidence E

$P(E|H_i)$ = the probability that the evidence E is observed given that hypothesis i is true

$P(H_i)$ = the a priori probability that hypothesis i is true in the absence of any specific evidence. These probabilities are called prior probabilities or priors.

k = the number of possible hypotheses

The theorem then states that

$$P(H_i|E) = P(E|H_i) * P(H_i) / \sum_{n=1}^{k} P(E|H_n) * P(H_n) \qquad (7.1)$$

For a long time, the Bayesian model had been the only numerical approach to inference with uncertainty, since no quantification was introduced in the patterns of plausible reasoning. One of the best-developed uses of Bayes' theorem for AI problems is in the solution of pattern recognition or classification problems. Bayes' theorem can be modified to handle a variety of more complicated situations. But there are several drawbacks to the use of Bayes' theorem. It is often difficult to collect all the a priori conditional and joint probabilities required. Doing so would require accumulating a great mass of data. Doing so would also be very expensive. But worse, the data would be obsolete by the time they were collected. It is very difficult to modify the database of a Bayesian system because of the large number of interactions between the various components of it. Also, evaluating Bayes' formula to give an accurate estimate of the probability of a particular outcome must be disjoint. It cannot ever happen that two of them occur at once. This is often not the case. The accuracy of Bayes' formula also depends on the availability of a complete set of hypotheses. In other words, it must always be the case that one of the known hypotheses is true. For all of these reasons, Bayes' theorem does not appear to solve problems that arise in uncertain reasoning in real-world problems, although it does serve as the basis for some probabilistic AI systems, e.g., PROSPECTOR [Ref. 61].

2. Dempster-Shafer Theory

Several mathematical models of uncertainty, which depart from the usual probability approach, have been recently proposed, particulary, Dempster- Shafer belief theory [Ref. 62]. It is also called mathematical theory of evidence. A scheme for combining evidence which includes

uncertainty or ignorance was devised by Dempster and later formulated within a flexible representation framework by Shafer. It is more general than either a Boolean or Bayesian approach, providing a formal method for integrating knowledge derived from a variety of sources for use in perceptual reasoning. In this formalism, the likelihood of a proposition A_i is represented as a subinterval, $[s(A_i), p(A_i)]$, of the unit interval, $[0, 1]$. The evidential support for proposition A_i is represented by $s(A_i)$, while $p(A_i)$ represents its degree of plausibility, $p(A_i)$ can also be interpreted as the degree to which one fails to doubt A_i, $p(A_i)$ being equal to one minus the evidential support of $\tilde{}A_i$ (the symbol "~" is the Boolean NOT), i.e., the plausibility is the complement of the support for $\tilde{}A_i$. So, $p(A_i)$ is

$$p(A_i) = 1 - s(\tilde{}A_i)$$

The lower value, $s(A_i)$, represents the support for that proposition sets a minimum value for its likelihood. The upper value, $p(A_i)$, denotes the plausibility of that proposition and establishes its maximum likelihood. Support may be interpreted as the total positive effect a body of evidence has on a proposition, while plausibility represents the total extent to which a body of evidence fails to refute a proposition. The degree of uncertainty about the actual probability value for a proposition correspond to the width of its interval. So, the "uncertainty of A_i" is:

$$u(A_i) = p(A_i) - s(A_i)$$

Dempster's rule of combination requires that the knowledge sources be independent. The representation involves the assignment by a knowledge source of "probability masses". The mass allocated by a certain knowledge source to A_i is denoted $m(A_i)$. To clarify, assume that there is a set of n mutually exclusive and exhaustive propositions, such as that the target is of type $A_1, A_2, \ldots\ldots\ldots, A_n$. The method of evidential reasoning can assign a probability mass denoted as $m(A_i)$ to any of the original n propositions or to disjunctions of the propositions. For example, a disjunction is the proposition that the target is of type A_1 or A_2 (denoted as $A_1 \vee A_2$) and the mass assignment is denoted as $m(A_1 \vee A_2)$. There are (2^n-1) such general propositions (including all the possible disjunctions) that may be assigned mass, and the masses summed over all of these propositions must equal unity. It is noticed that this is a more general form of representation differs from the standard Bayesian approach in which probabilities are assigned only to the original n propositions, disjunctions are not considered. The representation to uncertainty θ is mass assignment to the disjunction of all the original propositions and is denoted by

$$m(\theta) = m(A_1 \vee A_2 \vee \ldots\ldots\ldots\ldots \vee A_n) \qquad (7.2)$$

Finally, mass can be assigned to the negation of a proposition. For example, the mass assigned to the negation of a A_1(the target is not type A_1) is denoted

$$m(\tilde{\ }A_1) = m(A_2 \vee A_3 \vee \ldots\ldots\ldots\ldots \vee A_n) \qquad (7.3)$$

The support $s(A_1)$ for the basic proposition that the target type is A_1 is just the mass associated with $A_1(s(A_1) = m(A_1))$. For a more complex proposition such as that the target is either type A_1, A_2 or A_3, is expressed as

$$s(A_1 \vee A_2 \vee A_3) = m(A_1) + m(A_2) + m(A_3) + m(A_1 \vee A_2) + m(A_1 \vee A_3) + m(A_2 \vee A_3) + (A_1 \vee A_2 \vee A_3) \quad (7.4)$$

The plausibility of a given proposition as mentioned before, is the sum of all masses not assigned to its negation. Alternatively, $p(A_i)$ can be computed by summing all masses associated with A_i and all disjunctions, including θ, that contain A_i. For example,

$$p(A_1) = m(A_1) + m(A_1 \vee A_2) + \cdots\cdots\cdots + m(\theta) \quad (7.5)$$

The use of these Shafer-Dempster techniques as they are known appears more complex than the use of the simple Bayesian process with its single set of probabilities and one of the difficulties in pursuing such an approach is to determine whether the extra complexity is justified by the results which can be expected. It is important to remember that an operator may have to make decisions based on the outcome of the identity process [Ref. 63]. We can easily see that an output such as, for example, " the probability that the detected aircraft is an enemy is at least 30% and could be 70% " seems more likely confuse than to clarify.

3. Fuzzy Logic

The objective of fuzzy logic is to modify (or "fuzzify") logic so that it applies directly to informal arguments. Fuzzy logic results from two stages of "fuzzification":

a. The introduction of vague predicates into the object language. This result in some form of multivalued logic.

b. Treating the metalinguistic predicates "true" and "false" as themselves vague or fuzzy.

The second stage is by far the most radical and controversial. Fuzzy logics have been imported into AI to deal with areas of vagueness and incomplete

information. Most expert systems, for example, are forced to take decisions when not all the facts pertaining to the decision are available. In such contexts it is natural to employ logics which, unlike classical logic, are suited to reasoning with such incomplete information. Non-monotonic logic has also been developed, largly by the AI fraternity itself, to deal with reasoning with incomplete information. Moreover, many concepts employed in natural language and AI are claimed to be "vague", and the necessity of reasoning with such concepts suggests that some "logic of vagueness" is appropriate [Ref. 64]. For example, the concept "young", it may be said that people under 10 years of age are young and those above 60 years are not young. However, there is no particular day at which a person's age switches from "young" to "not young", rather, this is a gradual transition. In fuzzy logic, the concept of young is expressed by a "membership function" representing the degree to which a person of a particular age can be considered to be young. It should be said that many applications of fuzzy logic are both philosophically and practically controversial, and the whole area is, at present, controversial.

Zadeh offers two main reasons for adopting fuzzy logic [Ref. 65,66]. First, he claims that it avoids the complexities introduced by regimentation of informal argument; secondly, he claims that it is the proper way to acknowledge that 'true' and 'false' are not precise but fuzzy.

In Fuzzy Logic (FL) the set of truth-values of the base logic, the set of points in the interval [0,1], is replaced by fuzzy subsets of that set. Zadeh does not, however, allow all fuzzy subsets. This, it is claimed, would result in 'unmanageable complexity'. Instead, Zadeh employs only a countable and

structured set of fuzzy subsets of [0,1] referred to as "linguistic truth-values". More explicitly, the Truth-Values (TV) set of FL is assumed to be a countable set, TV, of the form

$$TV = \{\text{true, false, not true, very true, not very true, more or less true, rather true, not very true, not very false, ...}\} \quad (7.6)$$

Each element of this set represents a fuzzy subset of [0,1]. Moreover, each element of TV is generated from the fuzzy set denoted by the term 'true'. So, for example, if U_{true} is the membership function of the fuzzy subset true, then the membership functions for the other members of TV might be given as follows:

$$U_{false}(v) = U_{true}(1 - v) \quad (7.7)$$

$$U_{nottrue}(v) = 1 - U_{true}(v) \quad (7.8)$$

$$U_{verytrue}(v) = (U_{true}(v))^2 \quad (7.9)$$

$$U_{rathertrue}(v) = (U_{true}(v))^{1/2} \quad (7.10)$$

etc., where v is the fuzzy variable. So that once the meaning of 'true', and the rules of computation are fixed, then so is the meaning of all the members of TV. As a consequence, the meaning of the linguistic truth-values (that is, the fuzzy subsets they denote) is crucially dependent upon the meaning chosen for 'true'. Moreover, it is quite difficult to see such a choice as anything other than arbitrary. Zadeh hints that the choice is motivated by the specific area of discourse under consideration. Consequently, the meanings assigned to the linguistic truth-values are localised.

How are the logical constants ~ , & , $\vee$ and $\rightarrow$ is used to obtain their meanings in a regime where truth values are elements of TV? As a first step we might proceed as follows:

$$[\sim A](v) \quad - \quad \neg([A](v)) \tag{7.11}$$

$$[A \,\&\, B](v) \quad - \quad [A](v) \wedge [B](v) \tag{7.12}$$

$$[A \vee B](v) \quad - \quad [A](v) \vee [B](v) \tag{7.13}$$

$$[A \rightarrow B](v) \quad - \quad [A](v) \rightarrow [B](v) \tag{7.14}$$

where the connectives ~ , & , $\vee$, $\neg$, $\wedge$ and $\rightarrow$ are those of the base logic and each [A] denotes a fuzzy subset of [0,1] represented above by its membership function. But there is a problem with this way of proceeding. We want each sentence in the language to denote not just an arbitrary fuzzy subset of [0,1] but rather an element of TV. Unfortunately, the above semantics offers no guarantee of this. Zadeh circumvents this difficulty by introducing the notion of a "Linguistic Approximation" (LA). Each fuzzy subset A of [0,1] has associated with it an element A^* of TV, it is called Linguistic Approximation (LA). This is expressed as

$$A^* = LA(A). \tag{7.15}$$

Unfortunately, there is not an obvious candidate for the notion of 'best' of such approximation, nor a general technique for computing 'good' ones. But whatever the merits of this notion, Zadeh employs it to provide the meanings of the logical constants as follows:

$$[\sim A](v) \quad - \quad LA(\lambda v.\, \neg([A](v))) \tag{7.16}$$

$$[A \,\&\, B](v) \quad - \quad LA(\lambda v.([A](v) \wedge [B](v))) \tag{7.17}$$

$$[A \vee B](v) \quad - \quad LA(\lambda v.([A](v) \vee [B](v))) \tag{7.18}$$

$$[A \rightarrow B](v) \quad - \quad LA(\lambda v.([A](v) \rightarrow [B](v))) \tag{7.19}$$

Now, the functions [A] and [B] associate with each sentence an element of TV, the set of fuzzy truth-values.

The introduction of fuzzy truth-values paves the way for a rather radical approach to inference. According to Zadeh, inference is only 'approximate'. Zadeh illustrates his notion of approximate reasoning by reference to examples of the form

a is small
a and b are approximately equal
---
b is more or less small

To illustrate, consider the statement

a is small

Under the administration of classical logic this proposition would be rendered true just in case a belongs to the set which constitutes the extention of the predicate small. In fuzzy logic, however, things are somewhat more involved. The predicate small is fuzzy, and proposition "a is small" is interpreted as the assignment of a fuzzy predicate as the value of a variable which corresponds to an implied attribute of a. More explicitly, this proposition would be interpreted as the assignment equation

Height(a) = small

where "Height" is the implied attribute. In equational terms the second premise of our example would be rendered as

(Height(a), Height(b)) = approximately equal

where the right-hand side represents a fuzzy subset of $[0,1] \times [0,1]$.

In general, then, a proposition of the form

$(a_1, \ldots\ldots, a_n)$ is C

is rendered as the assignment equation

$$R(a_1, \ldots\ldots, a_n) = C$$

where R is the implied attribute. For simplicity Zadeh writes this as

$$(a_1, \ldots\ldots, a_n) = C$$

The premises of our example thus constitute a pair of assignment equations of the form

$$a = \text{small}$$

$$(a, b) = \text{approximately equal}$$

and, in general, a collection of propositions $(ai_1, \ldots\ldots, ai_n)$ is Ci, $0 \leq i \leq n-1$ yield a set of equations

$$(ai_1, \ldots\ldots, ai_n) = Ci \qquad 0 \leq i \leq n-1 \qquad (7.20)$$

For Zadeh, approximate inference amounts to solving such systems of equations. As with equations in ordinary algebra, we can solve for any of the variables involved in the equations. As an illustration, solving for b in our example yields:

$$b = LA[\text{small} \circ \text{approximately equal}],$$

where $\circ$ is the composition of fuzzy relations, and is given by

$$U_{\text{small} \circ \text{approximately equal}}(b) =$$

$$\bigvee_x [U_{\text{small}}(x) \wedge U_{\text{approximately equal}}(x, b)] \qquad (7.21)$$

where $\bigvee_x$ represents the supremum over all objects in the domain of the fuzzy predicate "small".

Intuitively, the composition of the predicate "small" and the binary relation "approximately equal" represents the fuzzy predicate which returns that value which represents the best fit, between those objects which are in the domain of small, and which are approximately the same height as b.

According to Zadeh, the consequence of a given set of premises depends in an essential way on the meaning attached to the fuzzy sets which appear in the premises. This is, apparently, a consequence of the local character of fuzzy TV's. Consequently, validity can only be characterised semantically, and the traditional notions of completeness and consistency are peripheral to fuzzy logic.

In the light of the features of fuzzy logic, it would seem that it lacks the precise formal rules of inference in addition to the absence and apparent irrelevance of consistency and completeness results and the employment of a philosophically suspect theory of truth. All these, engender a feeling of insecurity. Indeed, as Haak points out [Ref. 67], fuzzy logic seems hardly recognisable as a logic at all. Probably, the best defence of fuzzy logic is located not in its conceptual foundations but in its potential applications [Ref. 68]. After all, many formal frameworks have been employed with much success even though their conceptual foundations have been in a sorry state.

4. Ad hoc Approaches

Many researchers in artificial intelligence have felt a need for alternatives of the standard Bayesian approach and have proposed and used, generally with success, more empirical models, particularly in expert systems such as MYCIN [Ref. 69], and others [Ref. 70,71]. Also many expert systems employ some form of numerical assignments to assertions which are often combined in ways which suggest that such assignments behave mathematically like probabilities.

One of the earliest approaches to reasoning with uncertainty was incorporated into the MYCIN system [Ref. 69]. It introduced a notion of " approximate implication" using numbers called " certainty factors" which were used to indicate the strength of a heuristic rule. For example, MYCIN's knowledge base includes the rule:

IF The infection is primary-bacteremia and the site of the culture is one of the sterile sites and the suspected portal of entry of the organism is the gastro-intestinal tract.

THEN There is suggestive evidence (.7) that the identity of the organism is bacteroides.

The number .7 is the certainty factor (in the range 0 to 1) of the conclusion. In MYCIN, assertions are not just true or false, the reasoning is vague or inexact and is indicated on a numerical scale. MYCIN's conjunction operator performs a minimisation, and its disjunction is furnished with a Bayesian interpretation. To elaborate, all assertions being considered by MYCIN have associated with them two numbers, a Measure of Belief (MB) and a Measure of Disbelief (MD). The MB of a hypothesis h given evidence e is the proportionate decrease in disbelief in h, and can be thought of in terms of probabilities as

$$MB[h,e] = \begin{cases} 1 & \text{if } P(h) = 1 \\ (\max[P(h|e),P(h)] - P(h))/(\max[1,0] - P(H)) & \text{otherwise} \end{cases} \quad (7.22)$$

Similarly, the MD is the proportionate decrease in belief in h as a result of e

$$MD[h,e] = \begin{cases} 1 & \text{if } P(h) = 0 \\ (\min[P(h|e),P(h)] - P(h))/(\min[1,0] - P(H)) & \text{otherwise} \end{cases} \quad (7.23)$$

A particular piece of evidence either increases the probability of h, in which case MB(h,e) > 0 and MD(h,e) = 0 (i.e., there is no reason to disbelieve h), or it decreases the probability of h, in which case MD(h,e) > 0 and MB(h,e) = 0. This relationship can be seen from the above formulas for MB and MD.

From these two measures, an overall estimate of the confidence of the system in its belief about the hypothesis can be computed. This estimate is called the Certainty Factor (CF) and is given as

$$CF[h,e] = MB[h,e] - MD[h,e] \tag{7.24}$$

It is noticed that if CF is positive, the system believes that the hypothesis is true; if CF is negative, there is more evidence against it and the system believes it to be false. By separating this measure into the two components MB and MD, the problem of slight confirmatory evidence being interpreted as disconfirmation is avoided. Considering several pieces of evidence, the measures of belief and disbelief of a hypothesis given two observations s_1 and s_2 are computed by:

$$MB[h,s_1\&s_2] = \begin{cases} 0 & \text{if } MD[h,s_1\&s_2] = 1 \\ MB[h,s_1] + MB[h,s_2]*(1-MB[h,s_1]) & \text{otherwise} \end{cases} \tag{7.25}$$

$$MD[h,s_1\&s_2] = \begin{cases} 0 & \text{if } MB[h,s_1\&s_2] = 1 \\ MD[h,s_1] + MD[h,s_2]*(1-MD[h,s_1]) & \text{otherwise} \end{cases} \tag{7.26}$$

One way to state these formulas in English is that the measure of belief in h is 0 if h is disbelieved with certainty. Otherwise, the measure of belief in h given two observations is the measure of belief given only one observation plus some increment for the second observation. This increment is computed by first taking the difference between 1 (certainty) and the

belief given only the first observation. This difference is the most that can be added by the second observation. The difference is then scaled by the belief in h given only the second observation. A corresponding explanation can be given, then, for the formula for computing disbelief. From MB and MD, CF can be computed. These formulas meet several requirements that one might wish them to satisfy, including commutativity, the order in which a set of observations is made is irrelevant.

A simple example will show how these functions operate. Suppose that an initial observation has been done that confirms our belief in h with MB=0.3. Then MD[h,e]=0 and $CF(h,s_1)$=0.3. Now a second observation has been done, which also confirms h, with $MB(h,s_2)$=0.2. Now

$$MB[h,s_1\&s_2] = 0.3 + 0.2*0.7 = 0.44$$

$$MD[h,s_1\&s_2] = 0$$

$$CF[h,s_1\&s_2] = 0.44$$

From this example it can be seen how slight confirmatory evidence can accumulate to produce increasingly larger certainty factors.

Sometimes it may be necessary to consider the certainty factor of a combination of hypotheses. It can be computed from the MB and MD of the combination. The formulas MYCIN uses for the MB of the conjunction and the disjunction of two hypotheses are

$$MB[h_1\&h_2,e] = \min(MB[h_1,e], MB[h_2,e]) \quad (7.27)$$

$$MB[h_1 \text{ or } h_2,e] = \max(MB[h_1,e], MB[h_2,e]) \quad (7.28)$$

MD can be computed analogously.

From tactical intelligence, the advantage of employing deterministic values instead of probabilities is often called for. For instance, an enemy will

most likely make a maximum g terminal maneuver rather than the average from a Monte Carlo simulation of all possible maneuvers. This tactical doctrine may prescribe a precise maneuver. Also, for very good reasons, people often feel uncomfortable estimating prior probabilities [Ref. 72]. Yet, they are willing to say whether a piece of evidence increases or decreases the probability of a hypothesis with respect to its prior value, and are often willing to use the certainty value to estimate the amount of change. Thus, certainties are particularly useful as a technique for talking about relative probabilities. In addition, they seem more natural than probabilities when establishing the context for a hypothesis.

By contrast to probability theory, these ad hoc approaches provide the expert with a language for more directly specifying how degrees of belief (expressed as subjective probabilities) are to be computed. The language does impose some constraints. It requires that functions for computing probabilities be composed out of small number of primitive functions. However, it also provides considerable freedom, such as allowing the specification of any loop-free network topology desired to group factors and control the flow of information. The price paid for this freedom is that there is no longer any guarantee that all of the axioms of probability theory will be honored. However, if one views the values computed as heuristic measures of degree of belief, then the only question is whether or not it is easy to construct an inference network that adequately approximates the specifications of the expert. A commonly voiced criticism of such approaches is that they are unnecessarily ad hoc. It is claimed that there are alternative approaches available which are better documented and understood [Ref. 72].

Mamdari and Efstathion [Ref. 73] for example, claim that fuzzy logic itself would provide more secure foundation for the enterprise. As a matter of fact, PROSPECTOR already employs some form of fuzzy-sets theory, at least according to the recent account given by Gaschning [Ref. 61].

Generally, it is hard to quantify and mathematically relate such subjective parameters as partial ECM, intelligence levels, the threat and human experience. So, it is adequate and easy to use ad hoc approaches to approximate the specifications of the expert.

G. EXPERT SYSTEM DEVELOPMENT TOOLS

It is important to distinguish between expert systems and expert system development tools. As shown in Figure 7.1, expert systems are specific applications consisting of a knowledge domain and an inference engine. A knowledge domain is generally the human expertize on a particular subject. The inference engine is the reasoning software. To obtain advice from an expert system, the user poses a problem via a user interface. The inference engine accepts the request, reasons about the query and the knowledge stored in the knowledge domain, and responds to the user. Expert systems can be used to provide expert advice and solve problems using a given knowledge domain. When a user presents a particular problem to the expert system, it uses the available reasoning knowledge to infer some advice, which it then reports to the user.

An expert system development tool can take the form of an Artificial Intelligence (AI) language such as Lisp or Prolog, an expert system shell, or an integrated artificial intelligence environment. Shells, a higher level of development tool, facilitate the development of expert systems by providing

a generalised inference engine and the ability to create knowledge domains in any subject area. An integrated expert system environment provides all the capabilities of a shell as well as other tools, such as decision support software. Expert system shells or environments (shell supersets) that support rule representation consist of a rule set editor (also known as a rule set manager), an inference engine, and generally, a user interface. Actually, the user interface and the rule editor can be embeded in the inference engine. The developer of an expert system uses a rule set editor to write rules. The inference engine uses the contents of the knowledge domain to arrive at its solution or recommendation.

H. PAIRWISE CORRELATION FOR TRACK FUSION

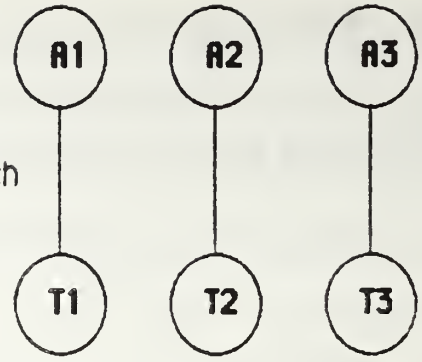
Track fusion is a difficult process and numerous algorithms have been defined for track correlation [Ref. 47, 63, 74, 75, 76, 77]. Many of these rely on probabilities to combine evidence, while others make hard yes or no decision. Because of the diversity of sensors that are operating in a large area surveillance system, the state vector coordinate system where the sensor/node level multitarget tracking problem is resolved will differ for the various local-track data bases. Assuming that the required coordinate transformation at each node is done using the appropiate transformation algorithm, the information collected by each node is overlaid in a common coordinate system. Furthermore, the time points at which the target states (latitude, longitude, etc.) are estimated may not coincide. Before fusing any two tracks, they are referred to a common time instant by using the prediction equations of the Kalman filter [Ref. 22].

Following Lakin and Miles [Ref. 9], we adopted pairwise correlations for track fusion. Figure 7.5, shows the steps used in pairwise correlations to solve the correlation ambiguity, which involves 3 distinct rule-driven steps :

1. First, assume all tracks (local and incoming) available at each node are separate and each track implies a new target.
2. Second, apply rules which create the possible pairwise correlations between each incoming track and existing local tracks. Those fail the pairwise correlations are considered new tracks for targets beyond the coverage of local sensor/sensors.
3. Third, apply rules to confirm strong correlations and to deny others. Where alternatives are of similar strengths, wait for further evidence.

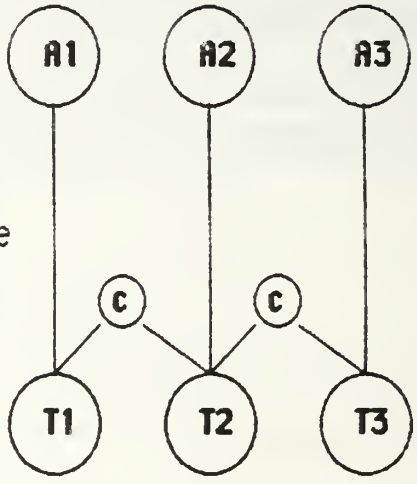
1. Step 1.

Assume all tracks available at each node are separate and each track implies a new target.



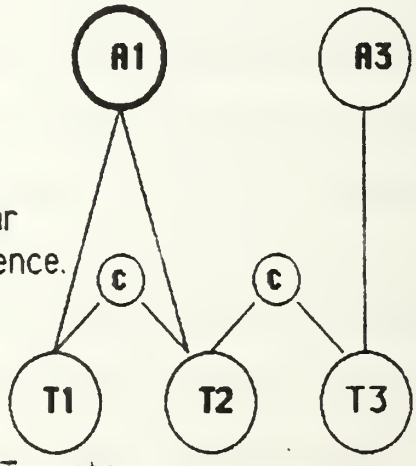
2. Step 2.

Apply rules which create the possible pairwise correlations between each incoming track and existing local tracks. Those fail the pairwise correlations are considered new tracks.



3. Step 3.

Apply rules to confirm strong correlations and to deny others. Where alternatives are of similar strengths, wait for further evidence.



T1, T2, T3 Targets Tracks

A1, A2, A3 Different Assumed Targets

C Pairwise Correlation Process

FIGURE 7.5. Pairwise Correlation for Track Fusion

The correlation process is basically based on the position and velocity closeness of each two tracks. The next logical step is to combine the correlated tracks to get the best estimate. A straightforward and a fairly obvious approach is taking a weighted average with most weight being given to the most accurate sensor. Another approach is using one track from a correlated pair or group (one which is belived the most accurate) as a representative track so that the other tracks are filtered out. One of the primary reasons for using this approach is that it serves the immediate purpose. If the accuracy available from a single sensor is sufficient to satisfy the needed requirements, then it is sensible to use it.

As it is mentioned before, the positional correlation process is a prerequisite for the fusion of identity and behavioural information. As noted earlier these types of information are of little practical use unless they are associated with position. When we correlate kinematic information we deal with dimensional data to which we can apply recognized mathematical tests for correlation. However, identity and behaviour cannot be treated the same way. Expert systems can be used, in which inference is performed using both sensor data and rules. Their structure allows the utilization of fully different kinds of information regardless of its form. This means that each information source is allowed to contribute information at its own level of detail. They process heuristic knowledge, apply logical inference, and reason with the human knowledge stored in the computer. Rather than facts, this knowledge represents the human 'rules of thumb' stored non-procedurally in the knowledge domain. Like humans they can reason about uncertain situations, factoring degrees of uncertainty into the reasoning

process. Certainty factors can be assigned and carried throughout the reasoning process, then reflected in the advice the expert system derives.

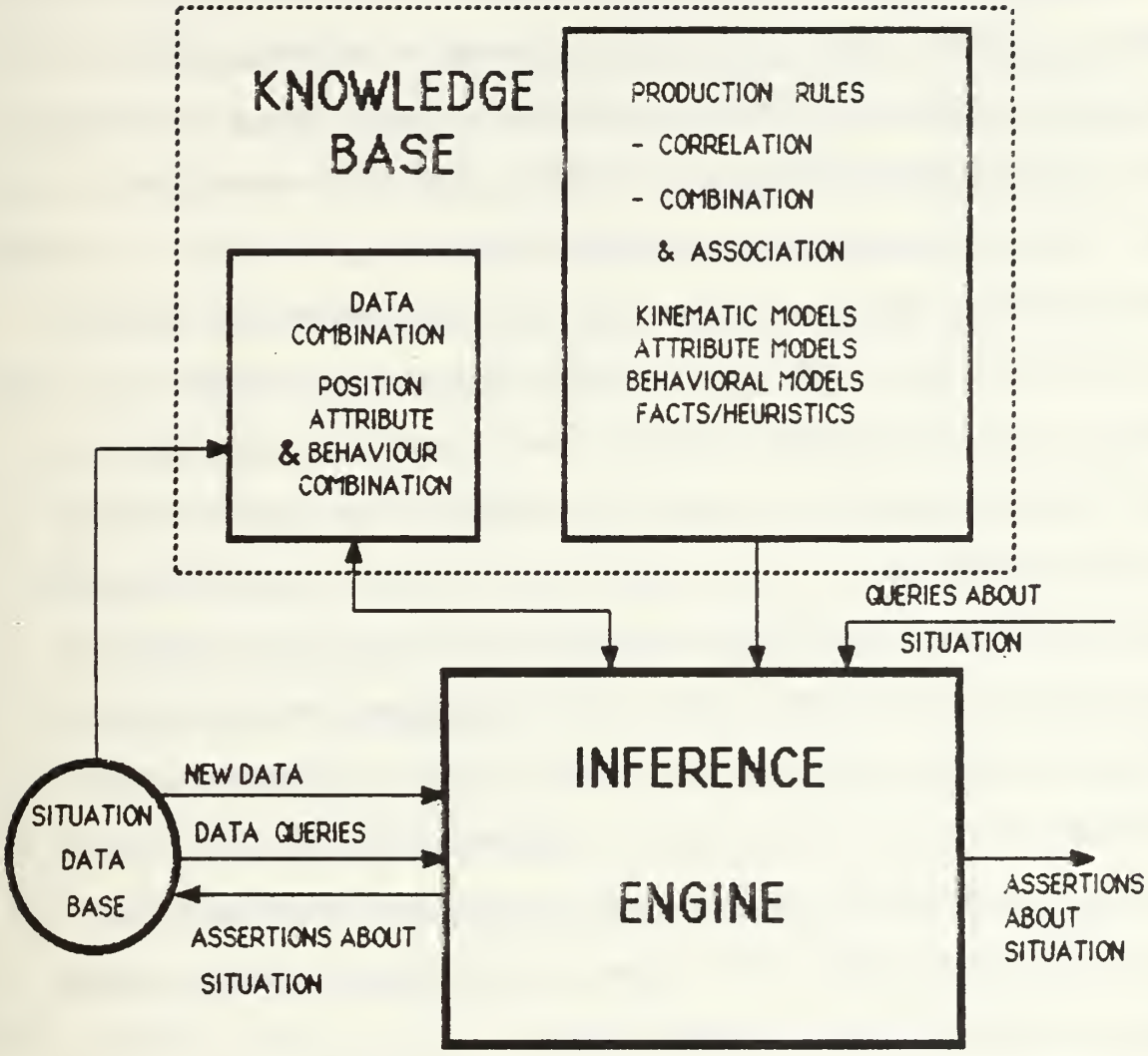


FIGURE 7.6. An Expert System Architecture For Track Fusion

VIII. THE SIMULATION SCENARIO

A. EXPERTS' VIEW OF MODERN RADAR ENVIRONMENT

Technically, computers are concerned only with those radar signals which relate to responses from aircrafts, and it is therefore unnecessary and uneconomic to feed them with all of the unwanted echoes which occur as a radar antenna sweeps through 360°, out to the range of its transmitted power. To be able to achieve this situation the radar signals of the concerned targets require to be converted into a digitised format. In this form the radar signals, radar data, or radar information can be fed into a computer, to be either displayed directly onto a radar display or, if the computer has the capacity or is linked to an additional computer containing the relevant flight plans and information, then both types of information can be correlated before being further processed onto the operator's display. A further key factor in this correlation, is the allocation of identity codes to individual aircrafts, through the use of Identification of Friend or Foe (IFF) for military aircrafts, or a Secondary Surveillance Radar (SSR) for civilian aircrafts. Also, there is the ability to enables those aircrafts which do not carry a transponder (SSR or IFF) to appear on the operator's display with a computer-related indication of their identity.

By allocating to the concerned aircraft a discrete IFF or SSR code, and also by informing the computer that the code so allocated refers to the specific flight information relevant to the aircraft, the computer is able to recognize

the radar information which it receives, and to correlate this information with the flight information already in its possession. The act of relating flight information and radar information begins to open up wide horizons for the application of automation to the tasks of controlling the air space. The operator himself acts as a communicator, a navigator, a calculator and a predictor of future events. It is essential to recognise that the advantages in automation are in reducing the workload upon the operator and be applied primarily to those of the operator's functions which limit his capability to discharge his primary responsibility, which is that of a decision-maker.

As an example of the standard format of the type of information of the flight plan which are in general use are:

1. Aircraft type;
2. Aircraft callsign;
3. SSR or IFF code;
4. Aerodrome of departure;
5. The aerodome of destination;
6. The proposed route of flight;
7. Estimated Departure Time (EDT);
8. The estimated time at the Flight Information Region (FIR) boundaries;
9. Height or desired cruising level;
10. Aircraft's cruising speed;
11. Type of flight (e.g. military/scheduled/general aviation);

The radar viewing units have also changed dramatically from the original cathode ray tube. It is usual, in modern radar units to which automation is

being applied, to use a synthetic type of radar display. A modern radar display console, adds the following facilities to the conventional radars:

a. Position symbols and labels adjacent to position symbols of selected aircrafts displaying important flight plan information.

b. Trail dots, which appear behind the aircraft's symbol to indicate the track which it has been following.

c. Visual alarms for an aircraft emergency, special hi-jack code, or radio faliure, these are in the form of flashing symbols and labels.

d. Tabular areas, these are areas upon which can be displayed any information of interrst to the operator.

e. Synthetic map displays, such as the outline of airways, and air-routes, the coast lines, danger areas, etc. These maps are usually programmed within the consol's computer memory. Also there are facilities which exist for the operator to draw-in on his display a synthetic map for any special purpose, such as, for example, a military exercise area or a temporary prohibited area for an air display.

Figure 8.1, provides some idea of the type of information which can be presented on TWS radar displays as applied to modern ATC systems [Ref. 78, 79, 80,81]. For example, for satisfying the requirements of modern ATC systems, which must have instantly available information that is both accurate and reliable, the aircraft itself is able to co-operate with the ground based radar systems. That is, it can carry its own airborne equipment, known as a 'transponder', which is capable of communicating with the ground-based SSR system. The transponder is activated by pairs of pulses transmitted by a ground interrogator, and its reaction is to transmit a train of pulses on a different radio frequency to the SSR interrogator receiver on the ground. Because the transponder is not relying upon reflected energy from the aircraft to provide a radar echo, but is making a full-blooded reply

itself, this enables the transmitters on the ground to be of lower power and employ simpler and cheaper technology and also ensure a certainty of signal return, unaffected by weather or other clutter factors. Also the returning train of pulses from the aircraft can be coded to contain information pertinent to that specific aircraft such as, for example, the identity of the aircraft and the height at which it is flying. This factor gives the SSR receiver and its computer processor the ability to separate and identify different targets in a manner that the ordinary radar cannot do, and then be able to compute additional information such as the speed of the aircraft and its flight attitude, all without recourse to any radio telophony speech with the pilot, other than an initial request to select a special group of code numerals on his SSR select panel in the cockpit.

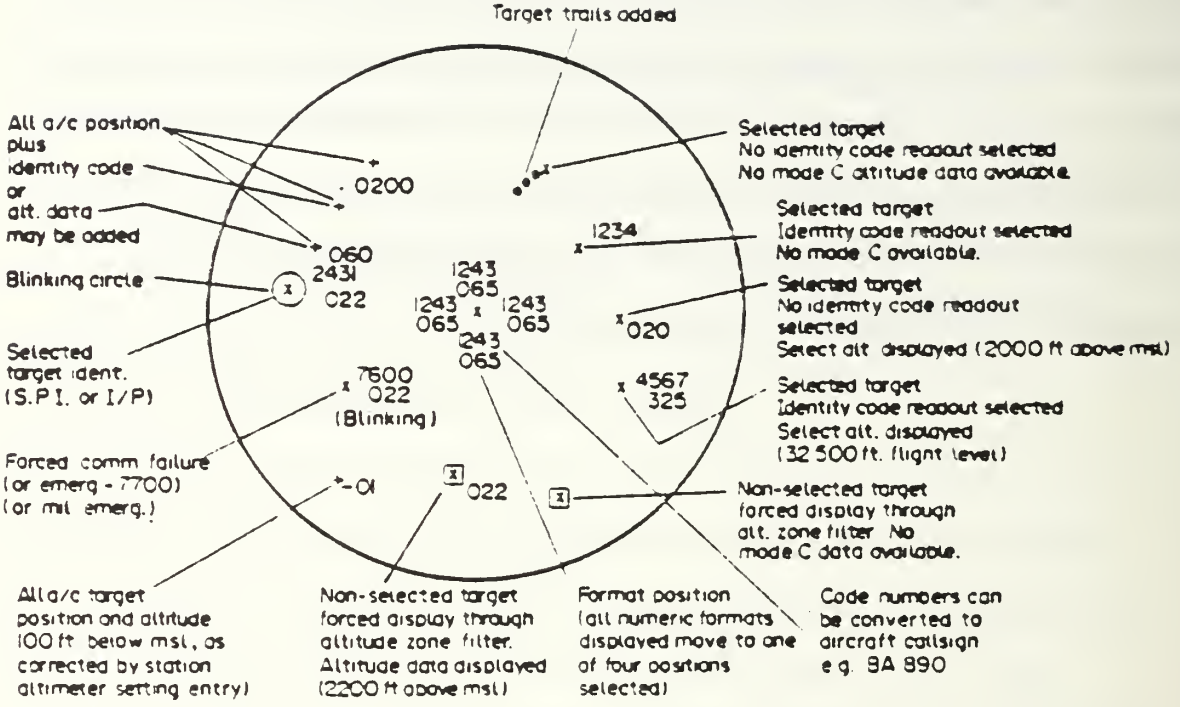


Figure 8.1. Modern ATC display with SSR Information

B. THE SCENARIO

The HEA approach is going to be applied on three different TWS systems **A**, **B**, and **C**. These systems are assumed to be tied together by appropriate direct communication links as shown in Figure 8.2. Each system has detected two different targets using its own radar sensor. The local data of each system gathered by its own radar sensor have been processed using its local estimation process, resulting in two well confirmed and distinct tracks. So, six tracks will result from the local estimators of the three TWS systems. These tracks are A1, A2, B1, B2, C1, and C2 from TWS systems **A**, **B**, and **C** respectively. The corresponding targets for these tracks are a1, a2, b1, b2, c1, and c2 for tracks A1, A2, B1, B2, C1, and C2 respectively. Using Equation (2.3), the network shown in Figure 8.2, is represented as

$$G = (V, E)$$

where $V = (\mathbf{A}, \mathbf{B}, \mathbf{C})$

and $E = ([\mathbf{A}, \mathbf{B}], [\mathbf{A}, \mathbf{C}], [\mathbf{B}, \mathbf{C}])$

The TWS systems together are assumed to cover a large air-space area with a partially overlapping fields of view. Each system is going to use locally its track information resulting from its local estimator (as that discussed in Chapter V), in addition to sending it to the the other two systems via the communication link used to tie each of the two other systems with the local one. By this way the area of coverage of each system is extended to cover the whole area covered by the three systems. This means that each system may get track information pertaining targets beyond the coverage of its local radar sensor. This can give the operator in each system's site an advance details on aircrafts which are due to enter his

sector of responsibility which seems to be of great help especially in ATC systems and C3 systems. It is assumed that the suitable network access protocol which secure error free exchange of track information is used, and the needed coordinate transformation and the referring to the common time instant as discussed in Chapter IV is done.

It is also assumed that the type of modern radar display console like that shown in Figure 8.1, is used at each TWS system. The fusion process is going to be performed in TWS system **A** based on its local track information and the other track information sent to it from TWS systems **B** and **C**. A pairwise correlation process is used to correlate each of the tracks A1, and A2 with the other tracks B1, B2, C1, and C2. The correlation process is going to be based on the kinematic information of the last report of each track (range, bearing and speed), in addition to its identity code, intent and behaviour. A simplified expert system approach as outlined in Chapter VII, using EXSYS expert system development package is used to perform the fusion process. The correlation process (embeded in the fusion process) will be basically started based on the position and velocity closeness of each two tracks by thresholding the absolute value of the difference between their position and velocity to a certain threshold value. This threshold value could be chosen based on the error covariance.

Two expert systems, TRAFUS1 and TRAFUS2 are developed. TRAFUS1 is a kind of an experimental hard decision knowledge-based track fusion approach. TRAFUS2 is a kind of an experimental soft decision knowledge-based track fusion approach. In these expert systems, simple inference rules are used to perform the correlation process and eliminate obviously

impossible pairing of tracks. The rules are allowed to be easily modified, added or deleted. New rules entered can be checked against the existing rules for consistency.

TRAFUS1 and TRAFUS2 obtain data needed to make a decision by asking the user questions relevant to the tracks needed to be fused. The user can also ask how the expert system reaches a decision. These two features are very helpful in training novice operators in real life applications. Almost no training is required to run any of them or any already developed expert system using EXSYS expert system development package. Details of TRAFUS1 and TRAFUS2 are presented in Chapter IX and Chapter X respectively. Their rules are described in Appendix E and Appendix F respectively.

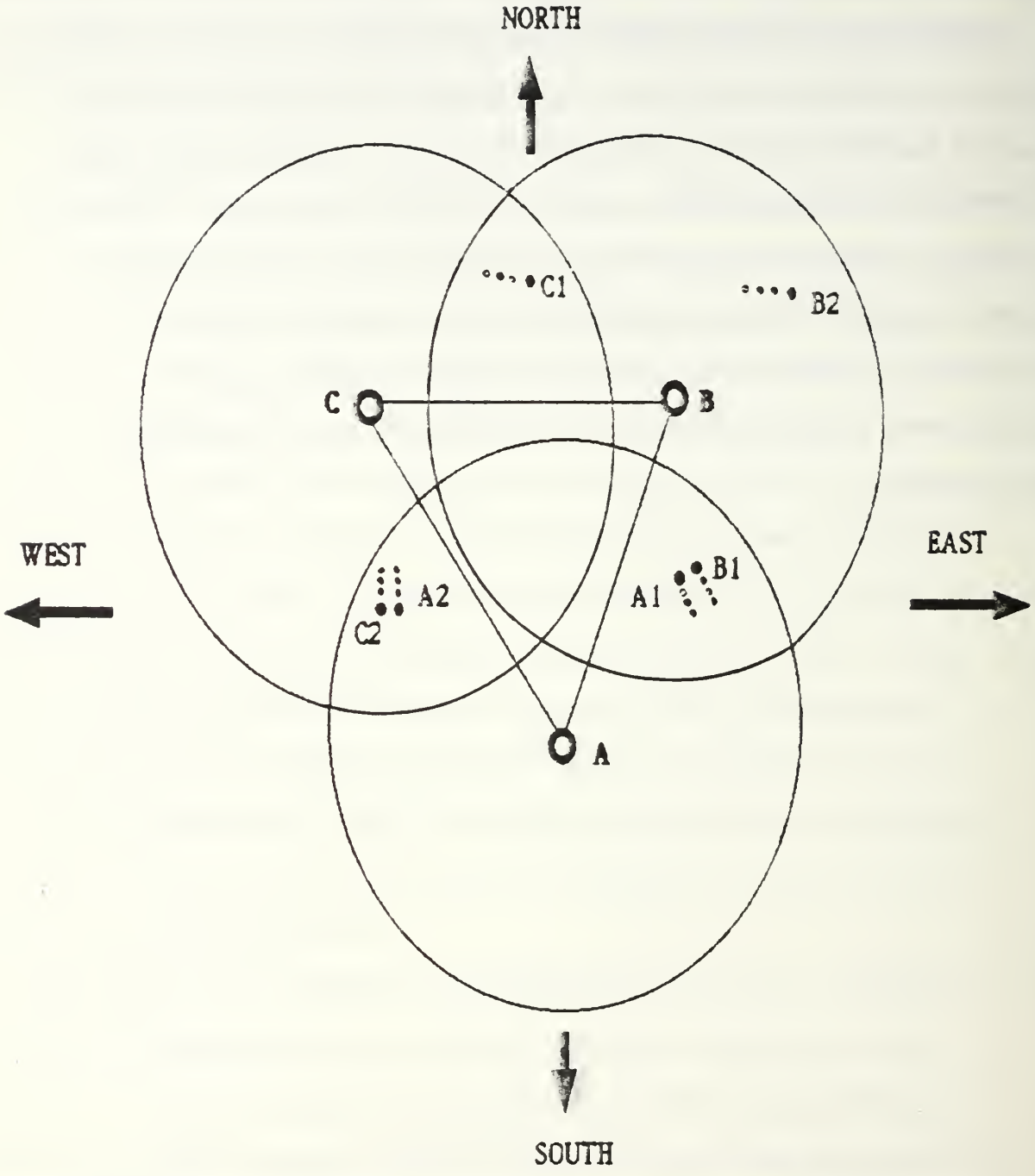


FIGURE 8.2. The Simulation Scenario

IX. TRAFUS1

A. RUNNING TRAFUS1

TRAFUS(from TRAck FUSion), refers to an experimental system to explore the applicability of artificial intelligence techniques to the implementation of an automated, extremely flexible track fusion consultant. To run TRAFUS1, the user enters EXSYS TRAFUS1 or just EXSYS and he will be asked for the filename which will be TRAFUS1. If TRAFUS1 is on other than the default drive, then he should enter the drive designator with the filename. For example, EXSYS B:TRAFUS1. Then he will be asked if he wishes instructions on how to use the program. After that he selects if he wishes to have rules displayed as the program runs. Then he will be asked questions relevant to the subject. The user answers by selecting one or more answers from a list or entering a numeric value. The expert system will continue to ask questions until it has reached a conclusion. The conclusion may be the selection of a single solution or a list of possible solutions arranged in order of likelihood. TRAFUS1 can explain, in English, how it arrived at its conclusion and why. If possible, the program will derive information from other rules rather than asking the user. This ability to derive information allows the program to combine many small pieces of knowledge to arrive at logical conclusions about complex problems. The rules editor of EXSYS allows the rules to be easily modified, added or deleted. All knowledge base files for TRAFUS1 are kept in two parts: one with a .RUL filename extension and one with a .TXT filename extension. Both must be on

will start asking the user questions relevant to the subject area of the knowledge base. This is how the program obtains the data needed to make a decision. There are two types of questions he may be asked: multiple choice and numeric value.

Multiple choice questions will display a statement ending in a verb, followed by a numbered list of possible completions of the sentence. The user should enter the number or numbers of the choices correct for his situation and press the [ENTER] key. If more than one number is chosen, the numbers should be separated with a space or a comma. If numbers outside the range of the list are entered, the program will re-ask the question. In fact the user won't get past the question until he answers it.

The other type of information the user may be asked for is a numeric value. There will be an explanation of what information the program needs and a space to inter the value by typing it and pressing [ENTER]. The number can include a decimal point.

The expert system will continue asking questions. When the program has obtained enough information to determine that all the IF conditions in a rule are true, it will display the rule, (unless the user has opted to not have rules displayed as they are used). If the computer determines that any of the IF conditions in a rule are false, it will reject the rule and go to the next appropriate rule. An example of the results obtained from a run of TRAFUS1 is presented in Figure 9.1.

```
1     Tracks A1 and B1 belong to the same target
   1
2     Tracks A2 and C2 belong to the same target
   1
3     Tracks A1 and B2 belong to different targets a1 and b2
   1
4     Tracks A1 and C1 belong to different targets a1 and c1
   1
5     Tracks A1 and C2 belong to different targets a1 and c2
   1
6     Tracks A2 and B1 belong to different targets a2 and b1
   1
7     Tracks A2 and B2 belong to different targets a2 and b2
   1
8     Tracks A2 and C1 belong to different targets a2 and c1
   1
9     Track B2 is for a hostile aircraft
   1
10    Track C1 is for a hostile aircraft
   1
11    Track A1 is for a friendly aircraft or an airliner
   1
12    Track A2 is for a friendly aircraft or an airliner
   1
13    Track B1 is for a friendly aircraft or an airliner
   1
14    Track C2 is for a friendly aircraft or an airliner
   1
15    Track A1 is a BOMBER
16    Track A1 is a CIVIL A/C
   1
17    Track A2 is a FIGHTER
```

FIGURE 9.1. Results Obtained from a Run of TRAFUS1

B. ASKING ABOUT RULES

When a rule is displayed the user has the option of asking how TRAFUS1 knows a condition in the IF part is true. To do this he enters the line number of the IF condition. The expert system will respond with one of four responses:

1. The expert system will display the rule or rules that allowed it to derive the information. A rule used for derivation will have information about the condition the user are asking about in its THEN part. He can then continue asking how the expert system knew that rule's IF conditions were true and so on. If the user asks about a condition that is an algebraic expression, the values of each of the variables in the expression will be displayed. He may then asks how these values were derived by entering the number of the variable.

2. If the user asks the expert system how it knows a condition is true that it did not derive, but determined by asking him for input, it will respond that he told it the information was true.

3. The user can ask for information about a condition that is several conditions down in the list and which the expert system may not have yet tested. This can occur when the user asks the expert system WHY in response to its question. If this is the case, the program will respond that it does not yet know if the condition is true or not.

4. In certain situations where the expert system has just derived new information, it may tell the user that the condition he is asking about is false and the rule will be eliminated.

Rules may have references for the source of the knowledge (e.g. personal observation, book, article, etc.). If, when a rule is displayed, the user presses [R] the expert system will display the reference for the rule if it has one. When the user are finished examining the rule, by pressing [ENTER], the expert system will continue asking him questions.

C. USING "WHY"

If the user wonders why the expert system needs to know the information it is requesting, he can ask it by typing WHY, instead of making a selection from the list of values, and presses the [ENTER] key. The expert system will respond by displaying the rule it is trying to determine the validity of. He may now ask the expert system about the IF conditions or references as described before. Then he presses [ENTER] when he is finished examining the rule. The expert system may now have the question originally asked redisplayed or it may display another rule. If the later is the case, it is because the first rule displayed was being used only to derive information needed by the second, and the second is the rule actually being tested. TRAFUS1 will continue showing the rules it is using to derive information until it reaches the base rule it is trying to apply. This rule will have at least one choice in its THEN part. By pressing the [ENTER] key, the program will be continued. If more than one rule was displayed, each time the user presses [ENTER] he will go one rule up the list being used in the derivation. He will then be reasked the question he responded to with "WHY".

D. ASKING HOW A CONCLUSION WAS REACHED

The user can ask the expert system how it arrived at its final value for a specific choice or why a statement is displayed. If he enters the line number for any choice or statement, TRAFUS1 will respond by displaying all of the rules it used to determine the value of that choice or statement. He again has all of the options in requesting more information about each of the rules as discussed above. If he wishes to learn why a choice not displayed was

eliminated by being given a probability value of 0, he presses [A] to have all choices displayed. Then he enters the line number of the choice in question.

E. CHANGING, RERUNNING AND PRINTING THE DATA

The user can easily test and analyze the effect his input had on the final outcome. He can change one or more of his answers, while holding the remainder constant, reruns the data and sees what effect the changes have on the final outcome. The current value for the choices can be saved for comparison with the new values. To change the data he presses [C]. He will be asked if he wishes to save the current values for comparison with the new ones he will be calculating. TRAFUS1 will then display a list of all of the information he the user provided by answering questions. Then he enters the number of the statement he wants to change and the expert system will reask that question. By answering the question with the new values that he wishes to try, the program will return to the display of all of the information that he told it. The user continues changing statements until the data is the way he wants it, then presses [R] to rerun the data. If, due to the changes, the program realizes that it needs more information, it will ask for it. The rules will not be displayed during the rerun. The program will then display the new list of choices. If he opted to have the previous values for comparison, they will be displayed in parenthesis.

He can change the data again in almost the same way. The only difference is that when he presses [C] he will be given three options:

1. Keep the original values for comparison.
2. Keep the most recently calculated values for comparison.
3. Don't keep any comparison data.

The ability to change and rerun the data allows the user to test the expert system and see if an answer that he were not sure of is vital to the final outcome, or really has little effect. He can save a printed copy of the results of the run by pressing [P]. Then he will be asked if he wishes to have the data he told the expert system also printed. If he presses [Y] he will have both the final sorted list of choices printed along with all of the data he provided the expert system in answer to its questions.

F. SAVING DATA AND RESULTS

The user has the option of storing the data he has input into TRAFUS1, exiting the program, and being able to return to the same point later. This can be useful if he needs to look up information needed by TRAFUS1 or if he must leave the program but don't want to loose the data he has already input. He can select to store the data by entering QUIT in response to any of the program requests for data. The program will then ask for the name of the file to store the data in. A filename of up to 8 characters (different than TRAFUS1) is needed to be entered. Then he will be asked if he wishes to return to the program or exit to DOS. Also he can store the input provided to reach the conclusions by pressing [Q]. This is the same as using the QUIT option when entering data. The data input will be stored in a disk file and he will be able to return directly to this point. This is particularly useful if he wants to experiment with the " change and rerun" command.

X. TRAFUS2

So far, in TRAFUS1, for the representation of facts and of methods for deducing new facts from old ones, we have almost always assumed that either a fact is known to be true, or it is known not to be true (or nothing at all is known about it). We have essentially not considered the possibility that we might know something that is " probably true ". However, there are situations in which such knowledge is important.

In TRAFUS2, An ad hoc approach is used, using the EXSYS 0-10 system. The systems uses numerical assignments to assertions, which expressing the degree of belief of the expert in these assertions. It is considered as a certainty factor which expresses the degree of confidence about a certain hypothesis. So,TRAFUS2 is generally similar to TRAFUS1, but the only difference is that the value following the "probability=" is a ratio where the denominator is 10. This is the most practical system. 0/10 is equivalent to "certainly false" and locks the value at 0/10 regardless of any other value the choice may have received. A value of 0/10 eliminates the choice from further consideration. A value of 10/10 is equivalent to "certainly true" and also locks the value for the choice at 10/10 regardless of any other values the choice may have received. Values of 1 to 9 represent degrees of certainty ranging from "very probably false" to "very probably true". The values from 1 to 9 do not lock the value and are averaged to give the final value for a choice.

For example, if a choice appears in three rules that had true IF parts with values of 3/10, 8/10, and 4/10, the final value for the choice will be the

average: 5/10. If the values found were 3/10, 9/10 and 0/10, the 0/10 would prevail and result in a final value of 0/10 regardless of other values. Likewise, if the values were 1/10, 3/10 and 10/10, the 10/10 would lock the value at 10/10 regardless of the previous lower values. Values of 1-9 are averaged to a final value only if not over-ridden by a 0/10 or 10/10. The first 0/10 or 10/10 prevails and will not be changed even by another 10/10 or 0/10.

In developing TRAFUS2, several questions have arisen and needed answers, among these questions are:

1. How to convert from human terms to numeric certainty factors. For example, what does " It is very likely that " mean?
2. How to normalize across different people's scales, particularly if the solution to question 1 is to get people to provide numbers directly.
3. How far to propagate changes in the Confidence Factor (CF) on the basis of new evidence. If the CF[h ,e] changes very slightly and h is part of the relevant evidence for another hypothesis, h, should CF[h,e] also be changed? If very tiny changes are always propagated as far as possible, the system may spend all of its time doing that with very little impact on the final outcome. On the other hand, many small changes can add up to a significant change that should not be ignored.
4. How to provide feedback to the database to improve the accuracy of the CF's of the rules. This problem has been particularly solved in MYCIN by the TEIRESIAS system's ability to explain the reasoning process to a physician and then to accept statements from the physician about how the rules should be revised.

For the solution of problems 1, 2, and 3, the word description of certainty shown in Figure 10.1, is used. For solving problem 4, the capability of EDITXS in EXSYS is used. Figure 10.2, shows the results obtained from a run of TRAFUS2.

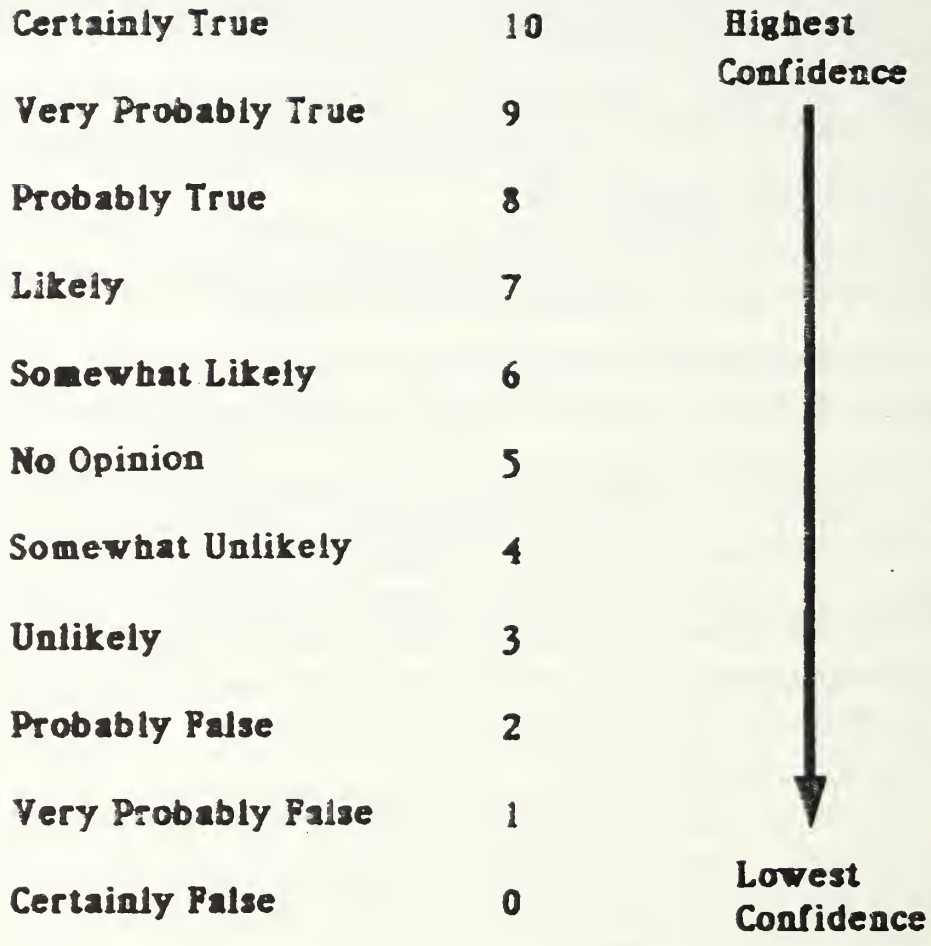


FIGURE 10.1. Word Description of Certainty in 0-10 System

	Values based on 0 - 10 system	VALUE
1	Tracks A1 and C1 belong to different targets a1 and c1	10
2	Tracks A1 and C2 belong to different targets a1 and c2	10
3	Tracks A2 and B1 belong to different targets a2 and b1	10
4	Tracks A2 and B2 belong to different targets a2 and b2	10
5	Tracks A2 and C1 belong to different targets a2 and c1	10
6	Tracks A2 and C2 belong to different targets a2 and c2	10
7	Track B2 is for a friendly aircraft or an airliner	9
8	Track A1 is for a friendly aircraft or an airliner	8
9	Track A2 is for a friendly aircraft or an airliner	8
10	Track B1 is for a friendly aircraft or an airliner	8
11	Track C2 is for a friendly aircraft or an airliner	8
12	Tracks A1 and B2 belong to different targets a1 and b2	8
13	Target a1 is a BOMBER	8
14	Target a1 is a CIVIL A/C	8
15	Target a2 is a BOMBER	8
16	Target b1 is a BOMBER	8
17	Target b1 is a CIVIL A/C	8
18	Target b2 is a BOMBER	8
19	Target b2 is a CIVIL A/C	8
20	Target c1 is a CIVIL A/C	8
21	Target c2 is a BOMBER	8
22	Track B2 is for a hostile aircraft	7
23	Track C1 is for a hostile aircraft	7
24	Tracks A1 and B1 belong to the same target	7
25	Targets a1 and b1 are the same target and they are one target	7
26	Target a2 is a CIVIL A/C	7
27	Target c2 is a CIVIL A/C	7
28	Range of last report of track A1 = 90.000000	
29	Range of last report of track B1 = 90.200000	
30	Bearing of last report of track A1 in degrees = 25.000000	
31	Bearing of last report of track B1 in degrees = 27.000000	
32	Speed of target a1 in m/s = 400.000000	
33	Speed of target b1 in m/s = 415.000000	
34	Identity of track A1 = ID345	
35	Identity code of track B1 = ID345	
36	Range of last report of track B2 = 250.000000	
37	Speed of track B2 in m/s = 200.000000	
38	Range of last report of track C1 in km = 230.000000	

FIGURE 10.2. Results Obtained from a Run of TRAFUS2

XI. CONCLUSIONS

In this dissertation, a new efficient and reliable distributed estimation architecture for Distributed Sensors Networks (DSN) has been presented. It is called Horizontal Estimation Architecture (HEA). HEA is introduced bearing in mind the technological advances in several disciplines, which are providing the future tools for designers of DSN, especially, the application of artificial intelligence and knowledge manipulation, and the adoption of decentralized decision making strategies in complex technological environments. In addition, the communication load is minimized between different nodes of a network by exchanging locally processed data between these nodes instead of raw data. Also, the HEA is developed bearing in mind the possibility of any hostile enemy actions, including physical destruction and electronic countermeasures which can create node and link failures and a dynamically changing network topology which are essential requirements for military systems.

A great motivation to HEA is the applicability of partitioning approaches which allow any large complex system to be divided into manageable proportions. The partitioning allows the usage of microcomputer systems, which provide a cost-effective solution for data processing. Obvious advantage of HEA as applied to DSN are local autonomy, heterogeneous feature, reliability, and survivability. Network splitting and reformation or connection of additional compatible networks are practicable during system operation and do not cause any restrictions as the new system is

initiated. The main concepts and features of HEA were presented in two conferences [Ref. 82, 83].

Through the integrated view of the assets of a DSN as shown in Figure 2.4, the HEA has been applied to TWS radar systems. This application shows the effect of the partitioning approach used by solving the multitarget tracking problem at the sensor (node) level using the appropriate local estimation algorithm. This guarantees the maximum utilization of the local resources of each radar site and takes advantage of the advanced techniques taking place in each major block of a radar system. When the multitarget tracking problem is solved separately for each individual node, a somewhat redundant view of the surveillance area will result, depending on the degree of overlapping coverages between the radars. The output of the local estimator is a group of different tracks. By solving the multitarget tracking problem separately for each individual sensor (node), the track estimates for the targets in the surveillance area become consolidated first at the level of each individual node.

A major component of HEA is the information fusion process. The information fusion process decides whether more than one track from different nodes represent the same target. A pairwise correlation technique is used for and proved to be easy to implement. The corresponding consistent tracks are combined together. Two techniques are used for information fusion. The first is based on algorithmic processing of track kinematic information, and it is proved that the fused estimates can be considered the global estimates of the tracks in the overlapping area assuming that the local estimates are optimal. It is also proved that there is

no need for the calculation of the cross covariance of the fused tracks since the local estimates and its associated covariance are the kind of information exchanged between different nodes. The second technique is based on heuristic reasoning by using expert systems to encapsulate target identity, behavior, intent and human expertise in computer software. An EXSYS expert system development package is used for this purpose. EXSYS employs AI techniques using currently available hardware and software. It does not require the complexity and cost of LISP driven architectures, nor is there is a need for large on-site support staff. The developed expert systems can be used mainly as an advisory tool for the manual operator. With modifications, it may directly control the fusion process autonomously.

Using the expert system approach, the correlation process can be easily implemented using simple production rules. The emphasis in the programs TRAFUS1 and TRAFUS2 has been on the fusion of track information, but fusion processes, especially in military applications, must integrally overlap with planning, ECM effects, tactical doctrine, operational limitations, logistics and historical reconstruction/analysis processes. The modification and augmentation of these into expert systems can be done.

Further research is needed for the application of HEA in systems which have different kinds of sensors. Also, many different kinds of knowledge engineering approaches are being applied to the various facets of information fusion problems. The data structures of the applicable expert systems vary greatly, and, in general, "talk" among these systems has not occurred. A considerable research effort is needed to establish a common ground for these systems to enable them to communicate with each other.

Another consideration about developing an expert system is tacit knowledge. Tacit knowledge has implications for knowledge elicitation within the current state of the art. One implication arises out of the invisibility of the relation between formal knowledge and skilled accomplishments, our lack of awareness of our own skills, and our frequently misplaced respect for theory-like representation of what we believe we know. The only solution to the problem demands that the knowledge engineer must do more than tap the knowledge of the expert, but must undertake at least a short apprenticeship, a period of participation as an observer, as part of the elicitation process.

APPENDIX A

THE SPECTRUM OF SENSORS AVAILABLE FOR DATA FUSION

Detectable Characteristics	Spectral Range	Sensor Systems
Acoustic Frequency	1 Hz - 10 kHz	Acoustic Detectors Active/Passive Sonar Seismometers
Electromagnetics Radio Frequency(RF)	1 Hz - 1 MHz (LF)	Magnetometers Passive ESM Receivers
	10 MHz - 100 MHz (HF/VHF/UHF) 1 - 10 GHz 10 - 50 GHz (SHF/EHF) 30 - 300 GHz (MMW)	Radar (Monostatic, Multistatic) - Surveillance - Fire Control Millimeter Radar Radiometers
InfraRed Wavelength (IR)	300 - 1□	IR Radiometers -Scanning IR Search Track -Focal Plane Arrays
Visible Light UltraViolet (UV)	0.7 - 0.4 □ 0.4 - 3x 10-3 □	Lazer Radar EO Sensors(TV) UV Spectrometers
Nuclear Paticles	3x 102 - 3x 10-4 A	X - Ray Detectors Gamma Ray Detectors
Non - Nuclear Particles		Mass Spectrometers

APPENDIX B

OVERVIEW OF TK!SOLVER

Computer-assisted problem- solving typically goes through the following steps:

Problem → Mathematical Model → Algorithm → Program → Results

The first step, based on the analysis of the problem, is to set up a mathematical model expressing the relationships between the variables. Then an algorithm (a precise description of how a computation is to proceed) must be developed for solving specific problems. Next a program implementing the algorithm is written in a conventional programming language (e.g. BASIC, FORTRAN, or PASCAL). Frequently, the programmer uses a diagram called a "flowchart" to facilitate the design and understanding of the algorithm or the program structure. A considerable amount of time is spent in these two stages, especially in the programming stage when the computer must be instructed in a step- by- step manner. Finally, of course, the program has to be debugged and run. All this means that a user has to "think like a computer" to solve a problem. This siphons a lot of time into other tasks necessary for fixing an unmatched parenthesis or formatting the output correctly. Solving a slightly different problem often means modifying the program and sometimes even reworking the algorithm.

TK!Solver shortcuts this problem-solving process by eliminating two steps, developing an algorithm and writting a program. Instead, the user inputs

the mathematical model directly using standard algebraic notation. By this way, the user and the computer interact at the level of mathematical models or relationships between variables and the computer automatically takes care of sequencing the operations. So, TK!Solver lets the user focus on the problem itself, not on how the problem is going to be solved using the computer. The "TK" in TK!Solver stands for Tool Kit, implying that there are several problem-solving tools in the kit and more to be expected. The exclamation mark refers to the Action kee, which is pressed to solve a problem or to make other things happen. TK!Solver solves problems for professionals working with numbers and formulas. It does this by processing equations entered in their natural form [Ref. 84].

Considered as an expert system [Ref. 84], figure B.1. shows the architecture of TK!Solver. The domain specific knowledge responsible for the high performance of the system is contained in the knowledge base. The problem solving tools embodying the control strategy, the direct and iterative solvers, utilize the knowledge base in the process of solving particular problems. For interaction (or I/O), TK!Solver provides "sheets" displayed through one or two windows on the screen. The main feature of the architecture is the explicit division between the knowledge base and the control strategy. Consequently, the expert/user deals only with issues of domain specific knowledge, and is insulated from the details of the implementation of the control strategy.

In the following paragraghs, the four components of the knowledge base, the characteristics of a model and the problem-solving mechanism are described:

1. Rules: The rule is the basic component of the domain-specific knowledge. It expresses the underlying mathematical relationship in terms of the equality of left-hand and right-hand side expressions. Equations, constraints, or definitions may all be represented as rules. The set of rules can be represented in the form of a network of relationships called R-graph (for relationships graph). A variable is represented by a node in the R-graph and each subgraph or polygon corresponds to a rule in the knowledge base.

2. Unit Conversions: Units of measurement are associated with most measurable quantities. Conversions between them are frequently encountered in problem solving and have to be defined in the knowledge base. The unit conversion feature in TK!Solver simplifies the conversion between the different units of measurement.

3. User Functions: Empiric relationships between sets of variables are expressed in the form of user-defined functions.

4. Built-In Knowledge: Irrespective of the domain-specific knowledge, TK!Solver can solve problems involving basic arithmetic operations and a large variety of built-in mathematical functions. For example, trigonometric functions, hyperbolic functions, exponential function, root function, natural and decadic logarithms and circular or inverse trigonometric functions. A standard variety of these is supplemented by a few special ones like "element" for retrieving list components or "apply" for associating empiric functions with arguments.

5. Model: The model encompasses the first three components of the knowledge base in Figure B.1. (rules, unit conversions, user functions) as contained in the rule, variable, unit, and user function sheets. In more general terms, the model can be seen as a compact, high-level representation of structure, organization, and content of the domain knowledge. The composition of the model coupled with its elegant internal representation allow for a simple yet powerful control strategy. The model also serves as a user-friendly guide during the problem-solving process. The model usually reflects a certain part of the knowledge base in a particular discipline. The models may be easily merged by the subsequent loading of some or all of the knowledge base components into TK!Solver, in order to create larger models capable of addressing more complicated problems. There is also the concept of TK!Solver packs or sets of models from particular disciplines.

< model> : : = < rule >[{(newline)< rule >)]
This means that a model consists of at least one rule; each rule must start on a new line, and there may be any number of lines between the rules.
< rule > : : = < expression >=< expression >[< comment >]
This means that a rule is represented by two expressions linked by an equal sign and followed, optionally, by a comment.
< comment > : : = "< character string >
This means that a rule comment consists of a quotation mark (") followed by a string of characters.

6. Problem-Solving Mechanism: The direct solver is the workhorse of the problem-solving mechanism. In it lies the grace and power of TK!Solver. It manipulates the equations depending on the problem formulation and solves for the unknown. If an incosistency error or an illegal operand is detected, the solution process is terminated, and the rule causing the problem is flagged with the appropriate error message. Since the solution path depends on the problem formulation, the control strategy may be considered as forward chaining or data-driven. Whenever the direct solver cannot match the nature and complexity of a given problem, the iterative solver can be used. The heart of the iterative solver is a modified Newton-Raphson procedure which handles sets of simultaneous linear and nonlinear equations []. It can be either explicitly invoked or automatically called when the direct solver fails to produce a solution.

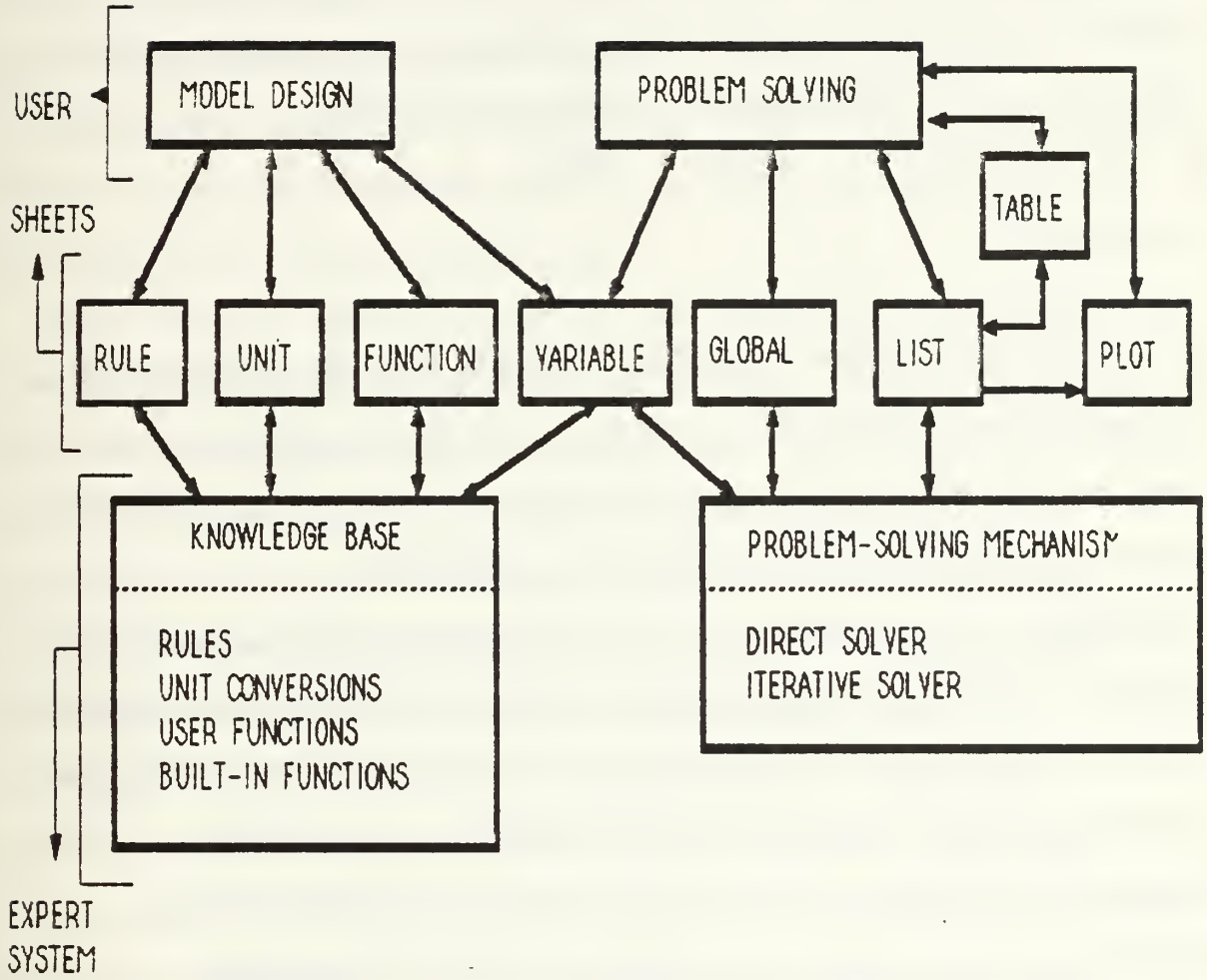


FIGURE B.1. Functional Diagram Of TK!Solver - User Interface

APPENDIX C

PRODUCTION RULES

In most areas of human decision making, the reasoning processes can be modelled by rule-based systems. A rule, known in these applications as a "production rule" or a "production", is generally of the form

IF F_1 and F_2 and ..... and F_n **THEN** C

or equivalently,

$$F_1 \,\&\, F_2 \,\&\, \cdots\cdots \,\&\, F_n \rightarrow C$$

where F_i is a fact, an event, a situation, a string of symbols, or a cause, and C is a conclusion or hypothesized conclusion, an action to be performed, or an effect. Some of the rules in a production system represent the knowledge of trained experts, and others provide system organization.

In addition to an organized set of rules, a production system must have a data base consisting, typically, of gathered pieces of evidence which might be relevant to the condition in the left side of some rules. System organization is provided by several kinds of control mechanisms. An evaluation mechanism is needed to evaluate the left side of a rule based on the evidence in the data base. It is desirable to have a mechanism for augmenting and modifying the system. A production system also needs direction and weighting mechanism.

Figure C.1, is an illustration of a tree structure of a very simple production system. The AND arcs denote single productions (where multiple conditions must be satisfied for the conclusion to follow), and OR inputs are separate

productions. The "direction" mechanism of a production system relates to reasoning processes, where inferring and deducing new information from evidence can be considered opposite in direction from hypothesizing and then testing the hypothesis. One type of system direction is forward chaining or running; these systems start with input data and proceed up to conclusions. Backward chaining (running), or top down, start with hypothesized conclusions that are selectively generated and proceed to see if they are supported by the data base. Some systems use an ad hoc combination of up and down directions.

When using a production system, there is often associated with each F_i in the premise a quantity known as a "certainty factor", which indicates the likelihood that F_i is true based on the input data. Also, for most production rules, the premise leads to the conclusion with , say, an 80% or 90% probability, instead of absolutely true or false. Similarly, there may be significant probability that the conclusion is true even when the premise is not satisfied. Measures of the latter two likelihoods are known as "strengths", "attenuation factors", or "certainty factors". All of these quantities can be used in estimating the certainty factor of a conclusion. Many conclusions are intermediate conclusions that are then treated as facts for future productions. "Weighting" is a term that refers to these quantities and their propagation through the tree. Weighting can be used to determine the reliability of final conclusions and also to reduce the number of computations through the pruning of unlikely hypotheses. If the statistics of the process are known sufficiently, Bayesian weighting can be used. A more common method of weighting is to use ad hoc scoring functions. When

the conditions F_i or the evidence about them cannot be considered independent, fuzzy set theory can be applied. For example, the fuzzy set computations $P(F_1, \ldots\ldots, F_n) = \min P(F_i)$ can be used at the AND nodes.

An advantage of a production systems is that it can be designed to provide high user confidence. The user can read the lists of rules and can question any conclusion, and the system can present to him the facts and logicleading to the conclusion. If he disagrees he can change the rules; with an appropriate mechanism for modification and augmentation. modular pieces of knowledge in the form of production rules can be added or changed without difficulty. In automated fusion applications, these system attributes are especially important. A user is unlikely to accept the system's conclusion if does not understand the logic behind it or previous conclusions. And he must be able to correct or refine the system and to incorporate new knowledge into the system when changes occur in hostile force procedures or equipment.

Aside from the obviously difficult task of acquiring rules, there are several special problems that will be encountered in applying production systems to fusion problems. At the system front-end there is the problem of evaluating the left side of a rule based on the conceptually structured data obtained through the processing of natural language reports. In platform identification applications, much of the data will be inaccurate or even totally wrong because of deception or human error. Moreover, conclusions will often have to be updated because of the continual arrival of data. There are also the geometrical problems of track association to be solved, but these

Probably the greatest problem with production systems or any automated system is that there are innumerable nonroutine situations which could occur. While a human might be able to fuse the data in an intelligent way in many of these situations, he probably would not be able to foresee the possibility of these situations in time to incorporate the necessary knowledge into an automated system. Generally, it seems that "blackboards" is a proper starting way for solving these problems.

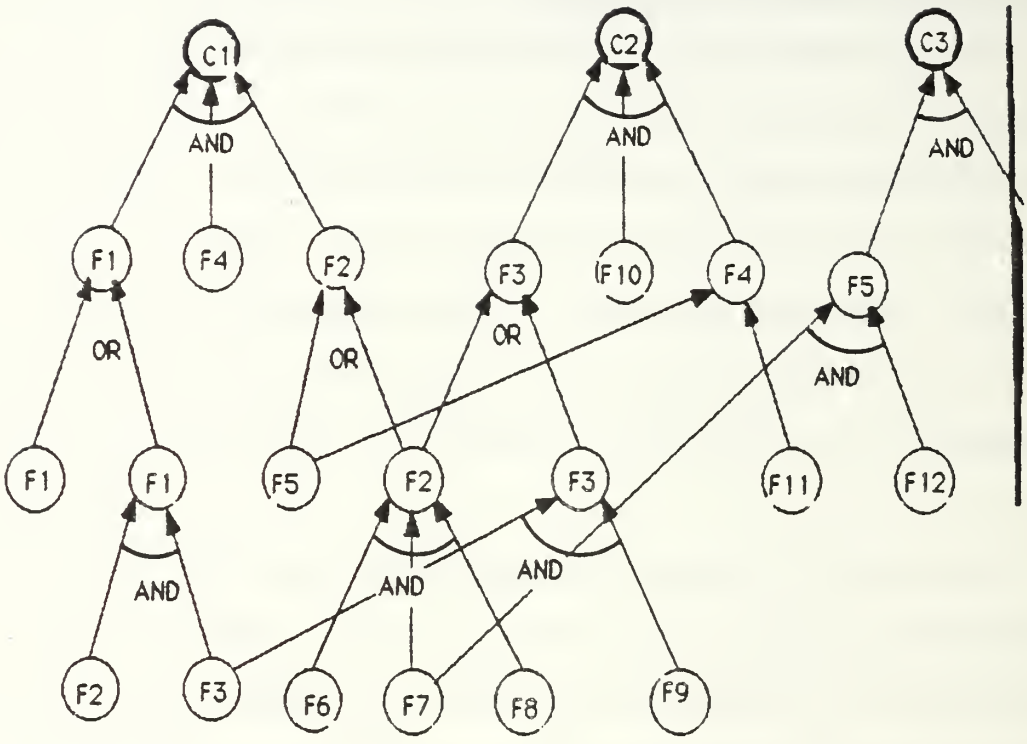


FIGURE C.1 Trees of Conclusion in a Production System

APPENDIX D

OVERVIEW OF EXSYS

EXSYS is a generalized personal computer expert system development package[Ref. 85]. It works on IBM PC, XT, AT or compatible. It can create about 700 rules, with an average of 6 or 7 conditions, per 64k of memory over 192k. That is about 5000 rules in a PC with 640k. The EXSYS programs are written in the C language producing small, fast running programs. The EXSYS development package consists of four main programs:

EXSYS.EXE: The runtime program for running existing expert systen knowledge bases

EDITXS.EXE: The program for generating, editing and testing your own expert system knowledge bases.

SHRINK.EXE: A utility program to compress the size of an edited knowledge base and rearrange the data in a knowledge base for rapid access.

FASTER.EXE: A utility program to rearrange the order of rules for maximum speed.

Expert systems work with knowledge to arrive at conclusions. This knowledge is in the form of rules that both the user and the computer can understand. The set of rules to solve a particular problem is often referred to as a knowledge base. A rule is divided into five parts, an **IF** part, a **THEN** part, an optional **ELSE** part, an optional **NOTE** and an optional **REFERENCE**.

IF

Condition

THEN

Conditions

ELSE

Conditions
and Choices

NOTE: .....

REFERENCE: .....

The **IF** part is simply a series of conditions, expressed as English sentences or algebraic expressions. The computer tests the conditions against the answers provided by a user and information that can be derived from other rules, to see if the **IF** conditions are true. The **THEN** part is also a series of conditions; however, there can also be choices with their associated probability values. The **ELSE** part is the same as the **THEN** part but is applied if any of the **IF** conditions are false. The **ELSE** part is optional and usually not needed in most rules. In some cases it is desirable to add a **note** to a rule to provide some special information to the user. If there is a **NOTE**, it will be displayed with the rule. The **NOTE** does not mean anything to the program, it is only for the user's information. The developer of the expert system knowledge base may also add a **REFERENCE** for a rule. This is intended to help the user find the source of the knowledge contained in the rule or more information if they should need it. The **REFERENCE** is displayed only if the user requests the reference; it is not automatically displayed with the rule.

To understand how the **EXSYS** expert systems are generated, it is needed to understand the definitions of the following terms:

1. Condition: In **EXSYS**, there are two main types of conditions, text and mathematical. For text, the condition is made up of two parts, a QUALIFIER and one or more VALUES. The qulifier is usually the part of the condition up to and including the verb. The values are the possible completions of the sentence started by the qualifier. A qualifier can have up to 30 possible values. If the developer finds that he needs more than 30 values, tries to divide them into groups each with less than 30 values. Then create a qualifier that selects among the groups and use it in conjunction with new qualifiers for each of the individual groups. When a new qualifier is created in **EXSYS** it is given a list of possible values such that combining the qualifier with a value (or values) makes a sentence. When more than one value is selected, the program will pu "or" between the values and, if any one of the listed values is true the condition will be true. A condition can also be formed by using a qualifier, "NOT" and one or more values. The selection of qualifier and values should be such that combining them with NOT, OR or AND makes a grammatically correct sentence. For mathematical conditions, they are represented as algebraic expressions. The mathematical expressions usually include VARIABLES. A variable is any string of alphanumeric characters, including spaces, enclosed in [ ]. The first 18 characters are significant but the variable can be up to100 characters. Only letters, digits and spaces can be used in variable names. In this case the variables are used to create evaluable algebraic expressions. An evaluable expression can be any algebraic expression from a single number to complex expressions. The following operators are recognized:

* (multiplication
/ (division)
+ (addition)
- (subtraction)
% (modulus operator)

Parentheses can be used to group expressions in the order of calculation the developer desires. Spaces can be included between operators to make the formula easier to read. The following functions are supported:

SIN( )
COS( )
TAN( )
ASIN( )
ACOS( )

ATAN( )
EXP( )
LOG( )
ABS( )
SQRT( )
INT( )

The trigonometric expressions are in radians. The log and exponential functions are base e. ABS is the absolute value. SQRT is the square root. INT is the integer part, with all fractions rounded down. The functions evaluate the expression in parentheses and perform the appropriate function on the result. The parenthetical expression must immediately follow the function name, without a space. A rule can have up to 126 conditions in each of its **IF**, **THEN** or **ELSE** part.

2. Choice: Choices are all the possible solutions to the problem among which the expert system will decide. The goal of **EXSYS** is to select the most likely choice based on the data input, or to provide a list of possible choices arranged in order of likelihood. The choices can be of any form, item, actions, etc., depending on what type of expert system is geing developed. EXSYS will display the text of the choice followed by "- Probability=" and a number. The number indicates the confidence that the choice is correct and is 0, 1 or a ratio such as A/B. The denominator, B, indicates the maximum possible value (either 10 or 100) in the calculational system being used. The numerator, A, is the probability value assigned to the choice. The person generating the knowledge base must select one of three options for how the program will use the probability data.

The rules are automatically invoked using backward chaining. **EXSYS** also supports forward chaining. External programs can be called by **EXSYS** for data acquisition and calculation and data can be passed back to **EXSYS** for analysis. This powerful feature enables it to handle a wide range of problems. The expert system can directly receive data from automatic testing equipment, data bases, some spread sheets and dedicated programs. There are two ways to call external programs from the expert system. The simplest is used to get data for a single variable or qualifier. The second

method is intended for obtaining data for a number of variables or qualifiers such as might be done with a data base, spread sheet or automatic test equipment. Both can easily be used with a wide range of programs and programming languages, including BASIC. There must be enough memory available for both **EXSYS** and the program called to run. **EXSYS** remains resident in memory while the called program runs.

Blackboarding is a powerful technique by which more than one knowledge base can share a common body of information. This can be done in **EXSYS** by having one one knowledge base write data to a file that is read by other knowledge bases. In version 3.1, special command options have been added to make this easier to do. There are two basic ways of controlling the order of execution of the various knowledge bases. The first is through the use of a batch file. This is appropriate if the knowledge bases are intended to be called in a specific, defined order. Batch files can include loops to restart the system. The second technique is to have an **EXSYS** expert system call other EXSYS expert system. This is the most powerful technique, as the order of excution of the knowledge bases can be varied depending on what is needed. Blackboarding requires care in making sure that the knowledge bases sharing the information each assign the data correctly, however, the benefits in being able to segment a problem and allowing extremely complex systems to be run in a PC make the technique quite worth while.

APPENDIX E

SAMPLE OF TRAFUS1 PRODUCTION RULES

Subject:
Hard Decision Knowledge-Based Multitarget Multisensor Track Fusion

Author:
Alaa Eldin M. Fahmy

Starting text:
This Expert System is an experimental system to explore the applicability of artificial intelligence techniques to the implementation of an automated, extremely flexible track fusion consultant. The expert system fuses tracks from three different TWS systems, A,B,C. Each system has two different tracks, resulting in a total of 6 tracks. The tracks are A1,A2,B1,B2,C1,C2 from TWS sites A,B,and C respectively. TWS sites are covering large air-space areas with partially overlapping fields of view. Track fusion is based on kinematic, attribute and behavier information.

Ending text:
The expert system will display which tracks are the same and to which target they are belong. In addition to whether targets are friendly or hostile targets.

Stops after first successful rule in data derivations.

```
RULES:

---------------------------------------


RULE NUMBER: 1

IF:
       ABS([RA1]-[RB1]) <= .15
  and  ABS([PHIA1]-[PHIB1]) <= 2
  and  ABS([SPEED OF A1]-[SPEED OF B1])   <= 20
  and  [IDA1] = [IDB1]


THEN:
       Tracks A1 and B1 are for the same target - Probability=1
  and  Targets a1 and b1 are one target


ELSE:
       Track A1 is for target a1 - Probability=1
  and  Track B1 is for target b1 - Probability=1
  and  Targets a1 and b1 are  different targets

---------------------------------------


RULE NUMBER: 2

IF:
       ABS([RA1]-[RB2]) <= .15
  and  ABS([PHIA1]-[PHIB2]) <=2
  and  ABS([SPEED OF A1]-[SPEED OF B2]) <= 20
  and  [IDA1] = [IDB2]


THEN:
       Tracks A1 and B2 are for the same target - Probability=1
  and  Targets a1 and b2 are one target


ELSE:
       Track A1 is for target a1 - Probability=1
  and  Track B2 is for target b2 - Probability=1
  and  Targets a1 and b2 are  different targets
```

```
RULE NUMBER: 8

IF:
       ABS([RA2]-[RC2]) <= .2
  and  ABS([PHIA2]-[PHIC2]) <= 3
  and  ABS([SPEED OF A2]-[SPEED OF C2]) <= 30
  and  [IDA2] = [IDC2]


THEN:
       Tracks A2 and C2 are for the same target - Probability=1
  and  Targets a2 and c2 are one target


ELSE:
       Track A2 is for target a2 - Probability=1
  and  Track C2 is for target c2 - Probability=1
  and  Targets a2 and c2 are different targets

---------------------------------------



RULE NUMBER: 9

IF:
       Tracks A1 and B1 are for the same target
  and  Tracks A1 and C1 are for the same target


THEN:
       Tracks A1, B1, and C1 are for the same target
  and  Targets a1, b1, and c1 are the same and they are one target

---------------------------------------



RULE NUMBER: 10

IF:
       Tracks A1 and B2 are for the same target
  and  Tracks A1 and C1 are for the same target


THEN:
       Tracks A1, B2, and C1 are  for the same target
  and  Targets a1, b2, and c1 are the same and they are one target
```

RULE NUMBER: 11

IF:
Tracks A1 and B1 are for the same target
and Tracks A1 and C2 are for the same target

THEN:
Tracks A1, B1, and C2 are for the same target
and Targets a1, b1, and c2 are the same and they are one target

---

RULE NUMBER: 12

IF:
Tracks A1 and B2 are for the same target
and Tracks A1 and C2 are for the same target

THEN:
Tracks A1, B2, and C2 are for the same target
and Targets a1, b2, and c2 are the same and they are one target

---

RULE NUMBER: 13

IF:
Tracks A2 and B1 are for the same target
and Tracks A2 and C1 are for the same target

THEN:
Tracks A2, B1, and C1 are for the same target
and Targets a2, b1, and c1 are the same and they are one target

---

RULE NUMBER: 18

IF:
Track B1 is for a target not filed in the flight plan
and Target of track B1 is not responding to IFF or SSR
and The position of track B1 is outside all airliner airways
and Track B1 is comming from WEST

THEN:
Track B1 is for a hostile aircraft - Probability=1
and Alert friendly forces - Probability=1

ELSE:
Track B1 is for a friendly aircraft or an airliner - Probability=1

RULE NUMBER: 19

IF:
Track B2 is for an aircraft not filed in the flight plan
and Target of track B2 is not responding to IFF or SSR
and The position of track B2 is outside all airliner airways
and Track B2 is comming from WEST

THEN:
Track B2 is for a hostile aircraft - Probability=1
and Alert friendly forces - Probability=1

ELSE:
Track B2 is for a friendly aircraft or an airliner - Probability=1

RULE NUMBER: 20

IF:
Track C1 is for a target not filed in the flight plan
and Target of track C1 is not responding to IFF or SSR
and The position of track C1 is outside all airliner airways
and Track C1 is comming from WEST

THEN:
Track C1 is for a hostile aircraft - Probability=1
and Alert friendly forces - Probability=1

ELSE:
Track C1 is for a friendly aircraft or an airliner - Probability=1

RULE NUMBER: 21

IF:
Track C2 is for a target not filed in the flight plan
and Target of track C2 is not responding to IFF or SSR
and The position of track C2 is outside all airliner airways
and Track C2 is comming from WEST

THEN:
Track C2 is for a hostile aircraft - Probability=1
and Alert friendly forces - Probability=1

ELSE:
Track C2 is for a friendly aircraft or an airliner - Probability=1

RULE NUMBER: 22

IF:
ABS([SPEED OF A1]) <= 50

THEN:
Track A1 is for a HELICOPTER

```
RULE NUMBER: 27

IF:
       ABS([SPEED OF C2]) <= 50

THEN:
       Track C2 is  for a HELICOPTER

----------------------------------------


RULE NUMBER: 28

IF:
          50 < ABS([SPEED OF A1]) <= 200

THEN:
       Track A1 is  for a CIVIL A/C

----------------------------------------


RULE NUMBER: 35

IF:
       200 < ABS([SPEED OF A2]) <= 400

THEN:
       Track A2 is for a BOMBER

---------------------------
```

QUALIFIERS:

1 Tracks A1 and B1 are
for the same target

Used in rule(s): 9 11

2 Tracks A1 and C1 are
for the same target

Used in rule(s): 9 10

3 Tracks A1, B1, and C1 are
for the same target

Used in rule(s): (9)

4 Targets a1, b1, and c1 are
the same and they are one target

Used in rule(s): (9)

17 Tracks A2 and B2 are

for the same target

Used in rule(s): 14 15

44 Track A2 is

for a target not filed in the flight plan
for a target filed in the flight plan

Used in rule(s): 17

47 Track A2 is

comming from WEST
comming from EAST
comming from NORTH
comming from SOUTH

Used in rule(s): 17

69 Track A1 is

for a HELICOPTER
for an INTERCEPTOR
for a BOMBER
for a CIVIL A/C
for a MISSILE

Used in rule(s): (22) (28) (34) (40)

VARIABLES:

1 R1
R1 is the range of track A1 at time t
Displayed at the end of a run

Used in rule(s):

2 R5
R5 is the range of track B1 at time t

Used in rule(s):

3 PHI1
PHI1 is the bearing of track B1 at time t

Used in rule(s):

4 PHI3
PHI3 is the bearing of track B1 at time t
Displayed at the end of a run

Used in rule(s):

5 ID1
ID1 is the identity of track A1
Displayed at the end of a run

Used in rule(s):

6 ID3
ID3 is the identity of track B1
Displayed at the end of a run

Used in rule(s):

FORMULAS:

1 ABS(R1-R5) <-

Used in rule(s):

2 ABS([R1]-[R5]) <- .15

Used in rule(s):

3 ABS([PHI1]-[PHI3]) <- 2

Used in rule(s):

4 [ID1] - [ID3]

Used in rule(s):

5 ABS([SPEED OF A1]-[SPEED OF B1]) <- 3

Used in rule(s):

6 ABS([R1]-[R7]) <- .15

Used in rule(s):

APPENDIX F

SAMPLE OF TRAFUS2 PRODUCTION RULES

Subject:
Soft Decision Knowledge-Based Multitarget Multisensor Track Fusion

Author:
Alaa Eldin M. Fahmy

Starting text:
This expert system is the same as TRAFUS1 except that the probabilistic reasoning is used. In this case as a degree of belief of a rule, a probabilty ratio (o/10 - 10/10) is attached to its then part. The expert system obtains the data needed to make a decision by asking the user questions relevant to the tracks needed to be fused. The user can ask the expert system how it arrived at its final decision. The user can easily test and analyze the effect his input had on the final outcome. Also, the user can ask why the expert system needs to know the information it is requesting.

Ending text:
The expert system will display which tracks are belonging to the same target and which are not. The results will be associated with a probability ratio as a degree of belief in it. If there are more than one solution, they will be displayed according to their relative likelihood. Whether the targets are friendly or hostile will be displayed in addition to its kind (helicopter, fighter, bomber, civil A/C, or missile)

Uses all applicable rules in data derivations.

RULES:

RULE NUMBER: 1

IF:

ABS([RA1]-[RB1]) <=.2

THEN:

Tracks A1 and B1 belong to the same target - Probability=6/10
and Targets a1 and b1 are the same target and they are one target - Probability=6/10

ELSE:

Track A1 belongs to target a1 - Probability=7/10
and Track B1 belongs to target b1 - Probability=7/10
and Tracks A1 and B1 belong to different targets a1 and b1 - Probability=7/10

NOTE:

RA1 and RB1 are in km.

REFERENCE:

Col. Fahmy's Ph.D. dissertation, Sep. 1987.

RULE NUMBER: 5

IF:
ABS([SPEED OF A1]) <=50

THEN:
Target a1 is a HELICOPTER - Probability=8/10

RULE NUMBER: 6

IF:
50 < ABS([SPEED OF A1]) <= 200

THEN:
Target a1 is a CIVIL A/C - Probability=8/10

REFERENCE:
Col. Fahmy's Ph.D. dissertation, Sep. 1987.

RULE NUMBER: 7

IF:
200 < ABS([SPEED OF A1]) <= 400

THEN:
Target a1 is a BOMBER - Probability=8/10

REFERENCE:
Col. Fahmy's Ph.D. dissertation, Sep. 1987.

RULE NUMBER: 24

IF:
ABS([RA1]-[RC1]) <=.2
and ABS([PHIA1]-[PHIC1]) <=3

THEN:
Tracks A1 and C1 belong to the same target - Probability=7/10
and Targets a1 and c1 are the same target and they are one target - Probability=7/10
and Tracks A1 and C1 belong to different targets a1 and c1 - Probability=4/10

ELSE:
Tracks A1 and C1 belong to different targets a1 and c1 - Probability=7/10

NOTE:
Bearing information of TWS system C is less accurate than that of TWS systems A and B.

REFERENCE:
Col. Fahmy's Ph.D. dissertation, Sep. 1987.

RULE NUMBER: 25

IF:
ABS([RA1]-[RC1]) <=.2
and ABS([PHIA1]-[PHIC1]) <=3
and ABS([SPEED OF A1]-[SPEED OF C1]) <=25

THEN:
Tracks A1 and C1 belong to the same target - Probability=8/10
and Targets a1 and c1 are the same target and they are one target - Probability=8/10
and Tracks A1 and C1 belong to different targets a1 and c1 - Probability=2/10

ELSE:
Tracks A1 and C1 belong to different targets a1 and c1 - Probability=7/10

RULE NUMBER: 53

IF:
ABS([RA2]-[RC1]) <=.2
and ABS([PHIA2]-[PHIC1]) <=3
and ABS([SPEED OF A2]-[SPEED OF C1]) <= 25
and [IDA2] = [IDC1]

THEN:
Tracks A2 and C1 belong to the same target - Probability=9/10
and Targets a2 and c1 are the same target and they are one target - Probability=9/10

ELSE:
Tracks A2 and C1 belong to different targets a2 and c1 - Probability=10/10

RULE NUMBER: 60

IF:
Track B2 is for an aircraft not filed in the flight plan
and Track B2 is for an aircraft not responding to IFF or SSR

THEN:
Track B2 is for a hostile aircraft - Probability=7/10

ELSE:
Track B2 is for a friendly aircraft or an airliner - Probability=7/10

REFERENCE:
Col. Fahmy's Ph.D. dissertation, Sep. 1987.

RULE NUMBER: 68

IF:
Track B1 is for an aircraft not filed in the flight plan
and Track B1 is outside all airliner airways
and Track B1 is outside all airliner airways

THEN:
Track B1 is for a hostile aircraft - Probability=8/10

RULE NUMBER: 69

IF:
Track B1 is for an aircraft not filed in the flight plan
and Track B1 is outside all airliner airways
and Track B1 is outside all airliner airways
and Track B1 is comming from WEST

THEN:
Track B1 is for a hostile aircraft - Probability=9/10

RULE NUMBER: 70

IF:
Track B2 is for an aircraft not filed in the flight plan
and Track B2 is for an aircraft not responding to IFF or SSR

THEN:
Track B2 is for a hostile aircraft - Probability=7/10

RULE NUMBER: 88

IF:
Track B1 is for an aircraft responding to IFF or SSR

THEN:
Track B1 is for a friendly aircraft or an airliner - Probability=8/10

RULE NUMBER: 89

IF:
Track B1 is inside any of airliner airways

THEN:
Track B1 is for a friendly aircraft or an airliner - Probability=6/10

RULE NUMBER: 90

IF:
Track B1 is comming from EAST or comming from NORTH or comming from SOUTH

THEN:
Track B1 is for a friendly aircraft or an airliner - Probability=9/10

RULE NUMBER: 91

IF:
Track B2 is for an aircraft filed in the flight plan

THEN:
Track B2 is for a friendly aircraft or an airliner - Probability=9/10

QUALIFIERS:

1 Track A1 is

for an aircraft filed in the flight plan
for an aircraft not filed in the flight plan
for an aircraft responding to IFF or SSR
for an aircraft not responding to IFF or SSR
outside all airliner airways
inside any of airliner airways
comming from EAST
comming from WEST
comming from NORTH
comming from SOUTH

Used in rule(s): 57 63 64 79 80 81 82

2 Track A2 is

for an aircraft filed in the flight plan
for an aircraft not filed in the flight plan
for an aircraft responding to IFF or SSR
for an aircraft not responding to IFF or SSR
outside all airliner airways
inside any of airliner airways
comming from EAST
comming from WEST
comming from NORTH
comming from SOUTH

Used in rule(s): 58 65 66 83 84 85 86

CHOICES:

1 Track A1 belongs to target a1

Used in rule(s): [1] [2]

2 Track A2 belongs to target a2

Used in rule(s):

3 Track B1 belongs to target b1

Used in rule(s): [1] [2]

4 Track B2 belongs to target b2

Used in rule(s):

5 Track C1 belongs to target c1

Used in rule(s):

6 Track C2 belongs to target c2

Used in rule(s):

VARIABLES:

1 RA1
Range of last report of track A1
Numeric variable
Displayed at the end of a run

Used in rule(s):						
	1	2	3	4	10	11
	12	13	24	25	26	32
	33	34				

2 RB1
Range of last report of track B1
Numeric variable
Displayed at the end of a run

Used in rule(s):						
	1	2	3	4	40	41
	42					

3 PHIA1
Bearing of last report of track A1 in degrees
Numeric variable
Displayed at the end of a run

Used in rule(s):						
	2	3	4	11	12	13
	24	25	26	32	33	34

4 PHIB1
Bearing of last report of track B1 in degrees
Numeric variable
Displayed at the end of a run

Used in rule(s):						
	2	3	4	40	41	42

12 IDB2
Identity code of track B2
String variable
Displayed at the end of a run

Used in rule(s): 13 45

13 RC1
Range of last report of track C1 in km
Numeric variable
Displayed at the end of a run

Used in rule(s): 24 25 26 51 52 53

14 PHIC1
Bearing of last report of track C1 in degrees
Numeric variable
Displayed at the end of a run

Used in rule(s): 24 25 26 51 52 53

15 SPEED OF C1
Speed of track C1 in m/s
Numeric variable
Displayed at the end of a run

Used in rule(s): 25 26 27 28 29 30
31 52 53

16 IDC1
Identity code of track C1
String variable
Displayed at the end of a run

Used in rule(s): 26 53

LIST OF REFERENCES

1. Ricci, Fred J. and Schutzer Daniel, U.S Military Communications, Computer Science Press, 1986.

2. Chong C. Y., Tse E., and Mori S., Distributed Estimation in Networks, Proc. ACC, San Fracisco, CA, 1983.

3. Chong C. Y., Hierarchical Estimation , Second MIT/ONR Workshop on Distributed Information and Decision Systems, NPGS, Monterey, CA, July 1979.

4. Castanon D. A., and Sandell N. R.,Jr., Distributed Estimation for Large-Scale Event-Driven Systems, in Control and Dynamic Systems, Vol. 22, pp. 1-45, Academic Press, 1985.

5. Tracker, E. C., and Sanders, C. W., Decentralized Structures for State Estimation in Large Scale Systems, Large Scale Systems Theory and Applications, Vol. 1, No. 1, pp. 39-49, Feb. 1980.

6. Chong, C. Y., Mori S., Tse E., and Wishner R. P., Distributed Estimation in Distributed Sensor Networks, Proc. of American Control Conference, Arlington, VA, 1982.

7. Grimaldi, R. P., Discrete and Combinatorial Mathematics, Addison-Wesly, 1985.

8. Sanders, Donald H. , Computers Today, Academic Press, Inc., 1979.

9. Lakin W. L., and Miles J. A., IKBS in Multisensor Data Fusion, International Conf. on C3, Conf. Public. No. 247, 16-18 April, 1985.

10. Bird, D. F., International Standerds in Military Communications, International Conf. on C3, Conf. Public. No. 247, 16-18 April, 1985.

11. Stallings William, Data and Computer Communications, Macmillan Publish. , 1985.

12. Hovanessian S. A., Radar Detection and Tracking Systems, Artech House, 1982.

13. Skolnik Merrill I., Introduction to Radar Systems, McGraw Hill, 1980.

14. Brookner, E., Development in digital radar processing, Trends and perspective in signal processing, pp. 7-23, January 1982.

15. Farrina A., and Studer F. A., Radar Data Processing, Vol. 1, Research Studies Press, 1985.

16. Field Arnold, International Airtraffic Control, Pergamon Press, pp. 88-102,1985.

17. Farina, A., Pardini, S., Survey of Radar Data Processing Techniques in Air- Trafic- Control and Surveillance Systems, IEE Proc., Vol. 127, No. 3, pp. 190-203, June 1980.

18. Blackman Samuel S., Multiple-Target Tracking with Radar Applications, Artech House, 1986.

19. Chang C. B., and Tabaczywski A., Application of State Estimation to Target Tracking, IEEE Trans. on Automat. Contr., Vol. AC-29, No. 2, February 1984.

20. Bar-Shalom Yaakov, Tracking Methods in a Multitarget Environment, IEEE Transact. on Automat. Contr., Vol. AC-23, No. 4, August 1978.

21. Fortmann, T.E.,Bar-Shalom Y., and Scheffe M., Multi-Target Tracking Using Joint Probabilistic Data Association, Proceedings of the 1980 IEEE Conference on Decision and Control, Albuquerque, NM, pp. 807-812, Dec.1980

22. Farrina A., and Studer F. A., Radar Data Processing, Vol. 2, Research Studies, 1986.

23. Reid Donald B., An Algorithm for Tracking Multiple Targets, IEEE Transact. on Automat. Contr., Vol. AC-24, No. 6, December 1979.

24. Cantrell B., Description of an α-β Filter in Cartesian Coordinates, Naval Research Laboratory Report 7548, Distributed by NTIS, AS 759-011, March 1973.

25. Schooler C. C., Optimal α-β Filter for Systems with Modeling Inaccuracies, IEEE Transact. on AES, Vol. AES-11, No. 6, pp. 1300-1306, November 1975.

26. Navarro A. M., General Properties of α-β and α-β-γ Tracking Filters, Physics Laboratory Report PHL 9977-02, Distributed by NTIS, N77-24347, January 1977.

27. Maybeck, Peter S., Stochastic Models, Estimation and Control, Vol. II, Academic Press, 1982.

28. Yannone, Ronald M., An Adaptive UD Factorized Kalman Filter For Real-Time Tactical Aircraft Track-While-Scan Systems, Proc. IEEE Conference on Decision and Control, pp. 358-364, 1985.

29. Kalman, R. E., A New Approach to Linear Filtering and Prediction Problems, Trans. ASME, Vol. 82D, pp. 35-50, 1960.

30. Battin, R. H., Astronautical Guidance, McGraw Hill, pp. 338-340, 1964.

31. Battin, R. H., and Levine, G. M., Application of Kalman Filtering Techniques in The Apollo Program, in Theory and Applications of Kalman Filtering, AGARD, Feb. 1970.

32. Agee, W. and Turner, R., Triangular Decomposition of a Positive Definite Matrix Plus a Symmetric Dyad, with Application to Kalman Filtering, U.S. Army, White Sands Missile Test Range, Technical Report, No. 38, October 1972.

33. Carlson, N., A Fast Triangular Formulation of the Square-Root Filter, AIAA Journal, Vol. II, No. 9, pp. 1259-1265, 1973.

34. Bierman, G. J., Factorization Methods For Discrete Sequential Estimation, Academic press, 1977.

35. Yannone, R. M., Use of Cholesky Square Roots Amidst the UD Factorization Implementation of Kalman Filters Real Time Airborne Tracking Systems, 23rd IEEE Conference on Decision and Control, December 1984.

36. Travassos, R. H., and Andrews, A., VLSI Implementation of Parallel Kalman Filters, AIAA Guid. and Cont. Conf., Advanced Avionics Session, San Diego, August 1982.

37. Kung, S. Y., Whitehouse H. J., and Kailath T., VLSI and Modern Signal Processing, Prentice-Hall, Inc., pp. 375-388, 1985.

38. Bucy, R. S., and Senne K. D., Nonlinear Filtering Algorithms for Vector Processing Machines, Comp. & Math. with Appl., Vol. 6, No. 3, pp. 317-338, March 1980.

39. Andrews, A., Parallel Processing of The Kalman Filter, Int. Conf. on Parallel Processing, pp. 216-220, August 1981.

40. Travassos, R. H., Parallel Kalman Filtering, ISI-04 Tech. Report, Oct. 1981.

41. Chen, C. T., Introduction to Linear System Theory, Holt., Rinehart and Winston, Inc., New York, pp. 35-49, 1970.

42. Singer, R. A. and Stein J. J, Sensor Data of Imprecisely Determined Origin in Surveillance Systems, Proc. of the IEEE Conf. on Decision and Control, Miami Beach, FL., pp.171-175, Dec. 1971.

43. Singer, R. A., Sea R. G., and Housewright K. B., Derivation and Evaluation of Improved Tracking Filters for Use in Multitarget Environments, IEEE Tras. Inform. Theory, Vol. IT-20, pp. 423-432, July 1974.

44. Jaffer, A. J. and Bar-Shalom Y., On Optimal Tracking in Multiple-Target Environments, Proc. of the Third Symp. Nonlinear Estimation and Its Applications, pp. 112-117, San Diego, CA, Sept. 1971.

45. Bar-Shalom Y., and Tse E., Tracking in a Cluttered Environments with Probabilistic Data Association, Automatica, Vol. 11, pp. 451-460, Sept. 1975.

46. Fortmann,T. e., Bar-Shalom Y., and Scheffe M., Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association, IEEE Journal of Oceanic Engineering, OE-8, No. 3, pp. 173-184, July 1983.

47. Bar-Shalom, Y., On The Track-To-Track Correlation Problem, IEEE Trans. On AC, Vol. AC-26, April 1981.

48. Gelb A., Applied Optimal Estimation, MIT Press, 1974.

49. Maybeck Peter S., Stochastic Models, Estimation, and control, Volume I, Academic Press, Inc., 1979.

50. Hills Frederick L., A Tactical Intelligence Processing System (TIPS), Proc. Of The ACC, pp. 445-451, June 18-20, Seattle, WA, 1986.

51. Hayes-Roth, Frederick, ed., Building Expert Systems, Addison-Wesly, 1986.

52. Engleman, C.,Berg C. H.,and Bischcoff M., KNOBS: An Experimental Knowledge Based Tactical Air Mission Planning System and A Rule Based Aircraft Identification Simulation Facility, Proc. 6th IJCAI, 1979.

53. Klahr, P., McArthur, D., and Narian, S., SWIRL: An Object-Oriented Air Battle Simulator, Proc. AAAI, 1982.

54. Weiss, Sholom M., and Kulikowski, Casimir A., A Practical Guide to Designing Expert Systems, Rowman & Allanheld Publishers, 1984.

55. Waterman Donald A., A Guide to Expert Systems, Addison-Wesly, 1986.

56. Erman, L. D., Hays-Roth F., The HEARSAY-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty, Computing Surveys, V.12, pp. 213-253, December 1980.

57. Nii H. P., and Feigenbaum E. A., Rule-Based Understanding of Signals, in D. A. Waterman and Hays-Roth F., Pattern-Directed Inference System, Academic Press, pp. 483-501, San Francisco, 1978.

58. Nii H. P., Feigenbaum E. A., Anton J. S., and Rockmore A. J., Signal-to Symbol Transformation: HASP/SIAP Case Study, AI Magazine, Vol. 3, No. 2, pp. 23-25, Spring 1982.

59. Delaney John R., Multi-System Report Integration Using Blackboards, Proc. of ACC, pp. 457-462, June 18-20, Seattle, WA, 1986.

60. Prade, H., A Synthetic View of Approximate Reasoning, Proc. 8th IJCAI, pp. 130-136, 1983.

61. Gaschning, J., PROSPECTOR: An Expert System for Mineral Exploration, In Introductory Readings in Expert Systems, Michie D. (ed.), pp. 47-65, Gordon & Breach, 1982.

62. Shafer, G., A Mathematical Theory of Evidence, Princeton University Press, 1976.

63. Wilson G. B., Some Aspects of Data Fusion, Inter. Conf. on C3, Conf. Public. No. 249, April 1985.

64. Barr Avron, and Feigenbaum Edward A., The Handbook of Artificial Intlligence, Addison-Wesley, 1982.

65. Zadeh, L. A., PROF: A Meaning Representation Language for Natural Languages, In Fuzzy Reasoning and its Applications, Mamdani, E. H. and Gaines, B. R. (eds.), Academic Press, 1981.

66. Zadeh L. A., Fuzzy Sets as a Basis for a Theory of Possibility, Fuzzy Sets and Syst. 1, pp. 3-28, 1978.

67. Haack, S., The Philosophy of Logics, Cambridge University Press, 1981.

68. Mamdani, E. H., and Gaines, B. R., Fuzzy Reasoning and its Applications, Academic Press, 1981.

69. Shortliffe, E. H., Computer Based Medical Consultation: MYCIN, New York: American Elsevier, 1976.

70. Weiss S. M., Kulikowski C. A., Amarel S., Safir A. , A Model-Based Method for Computer-Aided Medical Decision-Making, Art. Intellig.,11,pp. 145-172, 1978.

71. Ishizuka M., Fu K. S., Yoo J. T. P., Inexact Inference for Rule-Based Damage Assessment of Existing Structures, Proc. 7th IJCAI, pp. 837-842, Vancouver, 1981.

72. Turner Raymond, Logics for Artificial Intelligence, Ellis Horwood Limited,pp. 101-114, 1984.

73. Mamdari, E. H. and Efstathion, J., Fuzzy Logic, Proc. ACM Symp. on Expert Systems, Brunel University, 1982.

74. Waltz, E. L., Data Fusion For C3I Systems, In C3I Handbook, 1st Ed. 1986.

75. Kanyuck A. j., and Singer r. a., Correlation of Multiple -Site Track Data, IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-6, No. 2, March 1970.

76. Kenefic R. J., and Goulette P. L., Sensor Netting via the Discrete TimeExtended Kalman Filter, IEEE Transactions on Aerospace and electronic Systems, Vol. AES-17, No. 4, July 1981.

77. Chang C. B., and Youens L. C., Measurement Correlation for Multiple Sensor Tracking in a Dense Target Environment , IEEE Transactions on AC, Vol. AC-27, No. 6, Dec. 1982.

78. Erick Chr., Introduction on LORADS and ASDE, AGARD Conference Procc. No. 273, pp. 19-1 to 19-18, October 1979.

79. Stoddart D. L., Applications of Microprocessors in Air Traffic Control Systems, AGARD Conference Procc. No. 273, pp. 22-1 to 22-10, October 1979.

80. Woodall P. J., and Shagena J. L., Very Lightweight Air Traffic Management System Using an Elocctronic Scan Antenna, AGARD Conference Procc. No. 273, pp. 14-1 to 14-12, October 1979.

81. Field Arnold, International Air Traffic Control, Pergamon Press,pp. 42-68, 1985.

82. Fahmy Alaa M., and Titus H. A., Horizontal Estimation and Information Fusion in Distributed Sensors Networks, Proc. of International Symposium on Circuits and Systems, pp. 243-245, Philadelphia, PA, May 4-7, 1987.

83. Fahmy Alaa M., and Titus H. A., Horizontal Estimation for the Solution of Multitarget Multisensor Tracking Problems, Proc. of American Control Conference, pp. 2103-2108, Minneapolis, MN, June 10-12, 1987.

84. Konopasek Milos, and Jayaraman Sundaresan, TK!Solver Book: A Guide to Problem-Solving in Science, McGraw-Hill, 1984.

85. EXSYS, Inc.; Manual of EXSYS Expert System Development Package, Albuquerque, NM 87194, 1987.

BIBLIOGRAPHY

Alspach P. L., and Lobbia R. N., A Score For Correct Data Association in Multi-Target Tracking, 18th IEEE Conf. on Decision and Control, Dec. 12-14, Fort Landerdale, Florida, 1979.

Baheti R. S., Efficient Approximation of Kalman Filter for Target Tracking, IEEE Trans. on Aero. and Elec. Systems, Vol. AES-22, No. 1, pp. 8-14 January 1986.

Baird P. D. A., RF Simulation For Co-Ordination of ECCM, Inter. Conf. on Radar, Conf. Publication No. 155, pp. 460-462, 25-28 October 1977.

Bar-Shalom Yaakov, and Marcus Glenn D., Tracking with Measurements of Uncertain Origin and Random Arrival Times, 18th IEEE Conf. on Decision and Control, Dec. 12-14, Fort Landerdale, Florida, 1979.

Beam W. R., Automated Support Systems in Military Command and Control, Proc. of Expert Systems in Government Symposium, pp. 357-361, Mclean, Virginia, Oct. 22-24, 1986.

Bechtel R. J.,and Morris P. H., STAMMER: System for Tactical Assessment of Multisource Messages, Even Radar, Technical Document 252, Naval Ocean Systems Center, May 1979.

Bechtel R. J., Morris P. H., McCall D. C., and Kibler D. F., STAMMER2: System for Tactical Assessment of Multisource Messages, Even Radar, Technical Document 298, Naval Ocean Systems Center, Volumes 1 and 2, October 1979.

Berenji Hamid, and Lum Henry, Application of Plausible Reasoning to AI Based Control Systems, Proc. of ACC, pp. 1655-1661, June 1987.

Billetter D. R., Efficient Radar Control, Microwave Journal, January 1986.

Binias G., Computer Controlled Tracking in Dense Target Environment Using a Phased Array Antenna, Inter. Conf. on Radar, Conf. Public. No. 155, pp. 155-159, October 1977.

Bowman C. L., Maximum Likelihood Track Correlation for Multisensor Integration, 18th IEEE Conference on Decision and Control, Dec. 12-14, Fort Landerdale, Florida 1979.

Casner P. G., and Prengaman R. J., Integration and Automation of Multiple Co-Located Radars, Inter. Confer. on Radar, Conf. Publication No. 155, 25-28 October 1977.

Castella F. R., An Adaptive Two-Dimensional Kalman Tracking Filter, IEEE Trans. on Aero. and Elec. Systems , Vol. AES-16, No. 6, pp. 822-829, November 1980.

Castella F. R., Tracking Accuracies with Position and Rate Measurements, IEEE Trans. on Aero. and Elec. Systems, Vol. AES-17, No. 3, pp. 433-437, May 1981.

Chair Z., Varshney P. K., Optimal Data Fusion in Multiple Sensor Detection Systems, IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-22, No. 1, January 1986.

Chong C. Y., Mori S., and Chang K. C., Information Fusion in Distributed Sensor Networks, Proc. of American Control Conference,1985.

Chong C. Y., Chang K. C., and Mori S., Distributed Tracking in Distributed Sensor Networks, Proceedings of ACC, June 18-20, Seattle, WA, 1986.

Cochran J. G., Current Developments In The Design Of Air Defence Ground Radar Systems, Inter. Conf. on Radar, Conf. Publication No. 155, pp. 16-19, 25-28 October 1977.

Courtemonche A. N., A Rule-Based System for Sonar Display Analysis, Proc. of Expert Systems in Government Symposium, pp. 338-341, Mclean, Virginia, Oct. 22-24, 1986.

Cram L. A., Gilday M. R., and Rossiter K. O., Basic Radar Evaluation By Computer Simulation, Inter. Conf. on Radar, Conf. Publication No. 155, pp. 455-459, 25-28 October 1977.

Dillard R. A., New Methodologies for Automated Data Fusion, Technical Report 364, Naval Ocean Systems Center, September 1978.

Dillard R. A., Higher Order Logic for Platform Identification in a Production System, Technical Document 288, Naval Ocean Systems Center, October 17, 1979.

Dillard R. A., Experimental Tests of PTAPS Performance in Three Types of Production System Structures, Technical Document 385, Naval Ocean Systems Center, September 17, 1980.

Dillard R. A., A Platform-Track Association Production Subsystem, in Proc. of the Fourth MIT/ONR Workshop on Command and Control, June 1981.

Dockery J. T., The Use of Fuzzy Sets in the Analysis of Military Command, in Proc. 39th MORS, 1978.

Dunning B. B., Ambiguity Resolution in Contact Data Correlation, ACC, June 18-20, Seattle, WA pp. 452-456, 1986.

Farina A., and Pardini S., Track- While -Scan Algorithm In A Clutter Environment, IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-14, No. 5, pp. 769-779, Sept. 1978.

Farina A., and Pardini S., Multiradar Tracking System Using Radial Velocity Measurements, IEEE Trans. on Aero. and Elec. Systems, Vol. AES-15, No. 4, pp. 555-563, July 1979.

Farina A., and Studer F. A., Radar and Sensor Netting: Present and Future, Microwave Journal, January 1986.

Fitts J. M., Precision Correlation Tracking via Optimal Weighting Functions, 18th IEEE Conference on Decision and Control, Dec. 12-14, Fort Landerdale, Florida, 1979.

Forrest R. N., Three Position Estimation Procedures, Report Number NP 555-84-13, June 1984.

Galati G., and Farina A., Signal Processing Techniques for Surveillance Radar: An Overview, Microwave Journal, pp. 133-140, June 1985.

Handelman D. A., and Stengel R. F., An Architecture for Real-Time Rule-Based Control, Proc. of ACC, pp. 1636-1642, June 1987.

Hayes-Roth, B., The Blackboard Architecture: A General Framework for Problem Solving, Heuristic Programming Project Report No. HPP-83-30, Stanford University, May 1983.

Hayes-Roth, B., A Blackboard Model of Control, Heuristic Programming Project Report No. HPP-83-38, Stanford University, June 1983.

Hutchinson C. E., The Kalman Filter Applied to Aerospace and Electronic Systems, IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-20, No. 4, pp. 500-504, July 1984.

Jain R., Decisionmaking in the Presence of Fuzziness and Uncertainty, Proc. Special Symp. on Fuzzy Set Theory and Applic., Vol. 2 of Proc. 1977 IEEE Conf. on Decision and Control, pp. 1318-1323, 1977.

Jazwinski A. H., Adaptive Filtering, Automatica, Vol. 5, pp. 475-485, 1969.

Kandel Abraham, Fuzzy Mathematical Techniques with Applications, Addison-Wesley Publishing Company, Inc., 1986.

Keirsey D. M., Natural Language Processing Applied to Navy Tactical Messages, Technical Document 405, Naval Ocean Systems Center, Dec. 17, 1980.

Klemm R., Adaptive Pre-Whitening Filter, AGARD-CP-103, 15-18 May, 1972.

Kurien Thomas, and Washburn, Jr R. B., Multiobject Tracking Using Passive Sensors, Procc. of ACC, June 1985.

Lindgren A. G., Irza J., and Nardone S. C., Trajectory Estimation with Uncertain and Nonassociated Data, IEEE Transactions on Aerospace and Electronic System Vol. AES-22, No. 1, January 1986.

Marcus M., A Theory of Syntactic Recognition for Natural Language, MIT Press, 1980.

McAulay R. J.,and Denlinger E., A Decision - Directed Adaptive Tracker, IEEE Tranc. on Aerospace and Electronic Systems, Vol. AES-9, No. 2, March 1973.

Mehra R. K., Approaches to Adaptive Filtering, IEEE Transactions on Automatic Control, October 1972.

Michels K. M., An Expert System for Missile and Space Mission Determination, Proc. of Expert Systems in Government Symposium, pp. 2-13, Mclean, Virginia, Oct. 22-24, 1986.

Miller, Jr. J. T., and Berry J. P., Multisensor Utilization Investigation, IEEE Inter. Conf. on Radar, Publication No. 155, 25-28 October 1977.

Morefield C. L., Solution of Multitarget Multisensor Tracking Problems on the ILLIAC IV Parallel Processor , Anual Asilomar Conference on Circuits, Sysrems, and Computers, Nov. 22-24, 1976.

Morefield C. L., Decision Directed Multitarget Tracking, IEEE Conf. in Decision and Control, pp. 1195-1201, 1978.

Morley A. R., and Wilson A. S., Multiradar Tracking in a Multisite Environment, IEE Inter. Conf. on Radar, Publication No. 155, pp. 66-71, 25-28 October 1977.

Nahin P. J., and Pokoski J. L., NCTR Plus Sensor Fusion Equals IFFN or Can Two Plus Two Equal Five?, IEEE Trans. on Aero. and Elec. Systems, Vol. AES-16, No. 3, pp. 320-337, May 1980.

Nesline F. W., and Zarchan P., Decision Tree Filters - A Nonlinear Approach to Filtering for Fire Control APPLICATIONS, IEEE Conf. on Decision and Control , San Diego, Jan. 10-12, 1979.

Pal Sankar K., and Majumder D. K., Fuzzy Mathematical Approach to Pattern Recognition, Wiley Eastern Limited, 1986.

Popoli R., and Blackman S., Expert System Allocation for the Electronically Scanned Antenna Radar, Proc. of ACC, pp. 1821-1826, June 1987.

Porter D. W., and Engler T. S., Multi-Object Tracking via a Recursive Generalized Likelihood Approach, 18th IEEE Conference on Decision and Control, Dec. 12-14, Fort Landerdale, Florida, 1979.

Quigley, A. C., Holmes J. E., and Tunnicliffe R. J., Radar Track Extraction Systems, Proc. of AGARD No. 197, June 1976.

Ralph A. P., Non-Linear Tracking Using Doppler Information, Microwave Journal, June 1985.

Ramachandra K. V., Position,Velocity and Acceleration Estimates From The Noisy Radar Measurements, IEE Proc. Vol. 131, Part F, No. 2, pp. 167-168, April 1984.

Reid Donald B., An Algorithm for Tracking Multiple Targets, IEEE Transactions on Automatic Control, Vol. AC-24, No. 6, December 1979.

Roberts J. B. G., Simpson P., Merrifield B. C., Applying Digital VLSI Technology to Radar Signal Processing, Microwave Journal, January 1986.

Shimura M., An Approach to pattern Recognition and Associative Memories Using Fuzzy Logic, in Fuzzy Sets and their Application to Cognitive and Decision Processes, Academic Press, 1975.

Singer R. A., Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets, IEEE Trans., AES-6, pp. 473-483, 1970.

Singer R. A., and Kanyuck A. J., Computer Control of Multiple Site Track Correlation, Automatica, Vol. 7, pp. 455-463, 1971.

Singer R. A., Stein J. J., An Optimal Tracking Filter for Processing Sensor Data of Imprecisely Determined Origin in Surveillance Systems, IEEE Conference on Decision and Control, December 15-17, Miami Beach, Florida, 1971.

Smith F. W.,and Spain D. S., Multisensor Tracking of Re-Entry Vehicles, Proc. of IEEE Conf. on Decision and Control, pp. 575-578, Clearwater, FL, Dec. 1976.

Smith, P. and Benchler G., A Branching Algorithm For Discriminating and Tracking Multiple Objects, IEEE Trans. AC, Vol. AC-20, pp. 101-104, 1975.

Sorenson H. W., On the Development of Practical Nonlinear Filters, Information Sciences 7, pp. 253-270, 1974.

Stigter Leo, Experience With Automatic Tracking Systems Of The Royal Netherlands Navy, AGARD 78, 1978.

Tapley B. D., Schutz B. E., and Abusali P. A. M., A New Method For Enhancement of Data Separability and Data Classification in Multisensor - Multitarget Tracking Problems, Proc. of ACC, PP. 1056-1058, June 1985.

Thomas H. W., and Lafas C. C., Use of Aircraft- Derived Data to Assist in ATC Tracking Systems, Part 1: Accuracy and Theoritical Considerations, IEE Proc., Vol. 129, Pt. F, No. 4, pp. 281-288, August 1982.

Thomas H. W., and Lefas C. C., Use of Aircraft- Derived Data to Assist in ATC Tracking Systems, Part 2: Some Practical Tracking Filters, IEE Proc., Vol. 129, Pt. F, No. 5, pp. 359-365, October 1982.

Tickle G., Radar System Simulation And Performance Prediction, Inter. Conf. on Radar, Conf. Publication No. 155, pp. 451-454, 25-28 October 1977.

Tugnait J. K., Detection and Estimation for Abruptly Changing Systems, IEEE Conference on Decision and Control, 1981.

Verriest E., Friedlander B., and Morf M., Distributed Processing in Estimation and Detection, Procc. of the 18th IEEE Conference on Decision and Control, pp. 153-157, December 12-14, Fort Lauderdale, Florida, 1979.

Willner D., Chang C. B., and Dunn K. P., Kalman Filter Algorithms For a Multi-Sensor System, Procc. of IEEE Conference on Decision and Control, Clearwater, FL., Dec. 1976.

Wilson J. D., and Trunk G. V., Initiation of Tracks in a Dense Detection Environment, AGARD 78, 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Professor H. A. Titus, Code 62Ts Naval Postgraduate School Monterey, California 93943	3
4.	Distinguished Professor G. J. Thaler, Code 62Tr Naval Postgraduate School Monterey, California 93943	1
5.	Professor Rudolf Panholzer, Code 62Pz Naval Postgraduate Schoo Monterey, California 93943	1
6.	Professor N. M. Schneidewind, Code 54Ss Naval Postgraduate School Monterey, California 93943	1
7.	Professor M. F. Platzer, Code 67Pl Naval Postgraduate School Monterey, California 93943	1
8.	Department Chairman, Code 62Po Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
9.	Commander, Air force Academy Egyptian Airforce Academy Belbiss, Egypt	1

10. Chief, Research and Advancements Authority — 1
Egyptian Armored Forces Headquarter
Nacre City, Cairo, Egypt

11. Chief, Armament Authority — 1
Armament Authority of Egyptian Armored Forces
Koprey El-qubaa, Cairo, Egypt

12. Chief, Training Authority — 1
Egyptian Armored Forces Headquarter
Nasre City, Cairo, Egypt

13. Col. Alaa M. Fahmy — 13
4 Ali El-laithy Street, Ard El-golf
Heliopolis, Cairo, Egypt

14. Chief, Military Technical College — 1
Military Technical College
Koprey El-qubaa, Cairo, Egypt

Thesis
F183 Fahmy
c.1 Horizontal estimation and information fusion in multitarget and multi-sensor environments.